本资源来自数缘社区

http://maths.utime.cn:81

欢迎来到数缘社区。本社区是一个高等数学及密码学的技术性论坛，由山东大学数学院研究生创办。在这里您可以尽情的遨游数学的海洋。作为站长，我诚挚的邀请您加入，希望大家能一起支持发展我们的论坛，充实每个版块。把您宝贵的资料与大家一起分享！

数学电子书库

每天都有来源于各类网站的与数学相关的新内容供大家浏览和下载，您既可以点击左键弹出网页在线阅读，又可以点右键选择下载。现在书库中藏书 1000 余本。如果本站没有您急需的电子书，可以发帖说明，我们有专人负责为您寻找您需要的电子书。

密码学论文库

国内首创信息安全专业的密码学论文库，主要收集欧密会（Eurocrypt）、美密会（Crypto）、亚密会（Asiacrypt）等国内外知名论文。现在论文库中收藏论文 4000 余篇（包括论文库版块 700 余篇、论坛顶部菜单“密码学会议论文集”3000 余篇）。如果本站没有您急需的密码学论文，可以发帖说明，我们有专人负责为您寻找您需要的论文。

提示：本站已经收集到 1981－2003 年欧密会、美密会全部论文以及 1997 年－2003 年五大会议全部论文（欧密会、美密会、亚密会、PKC、FSE）。

数学综合讨论区

论坛管理团队及部分会员来源于山东大学数学院七大专业（基础数学、应用数学、运筹学、控制论、计算数学、统计学、信息安全），在数学方面均为思维活跃、成绩优秀的研究生，相信会给您的数学学习带来很大的帮助。

密码学与网络安全

山东大学数学院的信息安全专业师资雄厚，前景广阔，具有密码理论、密码技术与网络安全技术三个研究方向。有一大批博士、硕士及本科生活跃于本论坛。本版块适合从事密码学或网络安全方面学习研究的朋友访问。

网络公式编辑器

数缘社区公式编辑器采用 Latex 语言，适用于任何支持图片格式的论坛或网页。在本论坛编辑好公式后，您可以将自动生成的公式图片的链接直接复制到你要发的帖子里以图片的形式发表。

如果您觉得本站对您的学习和成长有所帮助，请把它添加到您的收藏夹。如果您对本论坛有任何的意见或者建议，请来论坛留下您宝贵的意见。

附录 A：本站电子书库藏书目录

http://maths.utime.cn:81/bbs/dispbbs.asp?boardID=18&ID=2285

附录 B：版权问题

前　言

电子计算机和通信网络的广泛应用，一方面为人们的生活和工作带来了极大的方便，但另一方面也带来了许多亟待解决的问题，其中信息的安全性就是一个突出的问题。密码技术是保证信息的安全性的关键技术。信息的安全性主要有两个方面，即信息的保密性和认证性。保密的目的是防止对手破译系统中的机密信息。认证的目的有两个：一是验证信息的发送者是真正的，而不是冒充的；二是验证信息的完整性，在传送或存储过程中未被窜改、重放或延迟等。信息的认证性和信息的保密性是信息的安全性的两个不同方面，认证不能自动地提供保密性，而保密也不能自然地提供认证功能。在现有的大部分教材中侧重于信息的保密性的研究，本书将兼顾信息的保密性和认证性这两个方面的研究，并行处理。本书以密码算法和协议为主线，对当前的热门话题诸如电子货币，零知识协议，基于身份的密码算法和协议，以及密钥托管技术等作了比较详细的论述。

Shannon 在 1949 年发表了"保密通信的信息理论"(见第二章文献[1])，将密码学的研究纳入了科学的轨道，但该篇论文当时并没有引起人们的广泛重视，直到 70 年代中期，人类开始步入信息时代时才引起了普遍的重视。那时密码学研究出现了两件引人注目的事情：一件是 Diffie 和 Hellman 发表了"密码学的新方向"(见第一章文献[5])一文，提出一种崭新的密码体制，冲破了长期以来一直沿用的私钥体制；另一件是美国国家标准局(NBS)公开征集，并于 1977 年正式公布实施的美国数据加密标准(DES)。这两个事件标志着现代密码学的诞生。

全书共分 11 章。第 1～3 章和附录介绍了密码学的基础知识，包括密码学的基本概念，密码学的信息理论基础，密码学的复杂性理论基础和本书中所用到的一些最基本的数学知识。第 4～12 章主要介绍了现有的有代表性的算法和协议，其中包括一些有代表性的私钥密码算法、一些有代表性的公钥密码算法、各种数字签名方案、一些典型的 Hash 算法、一些流行的识别协议、一些密钥分配和交换协议及密钥托管技术、电子货币以及一些特殊的密码协议。

本书是作者在长期从事科研和教学的实践基础上编写的。为了适应不同层次和不同专业的读者，我们对内容作了精心的安排，汲取了国内外现有文献中的精华。各章末附有注记和文献，介绍与该课题有关的资料和发展现状，以便感兴趣的读者进一步研究。

作　者

1998 年 9 月 10 日

目　录

第1章 引 论

密码学是一门古老而又年青的科学，它用于保护军事和外交通信可追溯到几千年前。在当今的信息时代，大量的敏感信息如病历、法庭记录、资金转移、私人财产等常常通过公共通信设施或计算机网络来进行交换，而这些信息的秘密性和真实性是人们迫切需要的。因此，现代密码学的应用已不再局限于军事、政治和外交，其商用价值和社会价值也已得到了充分肯定。

密码学的发展历史大致可划分为三个阶段：

第一个阶段为从古代到1949年。这一时期可看作是科学密码学的前夜时期，这段时期的密码技术可以说是一种艺术，而不是一种科学，密码学专家常常是凭直觉和信念来进行密码设计和分析，而不是推理证明。

第二个阶段为从1949年到1975年。1949年Shannon发表的"保密系统的信息理论"一文为私钥密码系统建立了理论基础，从此密码学成为一门科学，但密码学直到今天仍具有艺术性，是具有艺术性的一门科学。这段时期密码学理论的研究工作进展不大，公开的密码学文献很少。1967年Kahn出版了一本专著《破译者》(The Codebreakers)[1]，该书没有任何新的技术思想，只记述了一段值得注意的完整经历，包括政府仍然认为是秘密的一些事情。它的意义在于它不仅记述了1967年之前密码学发展的历史，而且使许多不知道密码学的人了解了密码学。70年代初期，IBM发表了Feistel和他的同事们在这个学科方面的几篇技术报告[2,3,4]。

第三个阶段为1976年至今。1976年Diffie和Hellman的"密码学的新方向"[5]一文导致了密码学上的一场革命。他们首次证明了在发送端和接收端无密钥传输的保密通信是可能的，从而开创了公钥密码学的新纪元。

本章主要介绍密码学的一些基本概念和一些有代表性的古典密码体制及其密码分析。

1.1 密码学的基本概念

密码学(cryptology)是研究密码系统或通信安全的一门科学。它主要包括两个分支，即密码编码学(cryptography)和密码分析学(cryptanalytics)。密码编码学的主要目的是寻求保证消息保密性或认证性的方法，密码分析学的主要目的是研究加密消息的破译或消息的伪造。

采用密码技术可以隐蔽和保护需要保密的消息，使未授权者不能提取信息。被隐蔽的消息称作明文(plaintext)，隐蔽后的消息称作密文(ciphertext)或密报(cryptogram)。将明文变换成密文的过程称作加密(encryption)，其逆过程，即由密文恢复出原明文的过程称作解密(decryption)。对明文进行加密操作的人员称作加密员或密码员(cryptographer)。密码员对明文进行加密时所采用的一组规则称作加密算法(encryption algorithm)，传送

消息的预定对象称作接收者(receiver),他对密文进行解密时所采用的一组规则称作解密算法(decryption algorithm)。加密和解密算法的操作通常都是在一组密钥(key)控制下进行的,分别称为加密密钥(encryption key)和解密密钥(decryption key)。

根据密钥的特点,Simmons[6]将密码体制分为对称和非对称密码体制(symmetric 和 asymmetric cryptosystem)两种。对称密码体制又称单钥(one-key)或私钥(private key)或传统(classical)密码体制,非对称密码体制又称双钥(two-key)或公钥(public key)密码体制。在本书中,我们采用私钥和公钥密码体制这两个术语。在私钥密码体制中,加密密钥和解密密钥是一样的或彼此之间容易相互确定。按加密方式又可将私钥密码体制分为流密码(stream cipher)和分组密码(block cipher)两种。在流密码中,将明文消息按字符逐位地加密。在分组密码中,将明文消息分组(每组含有多个字符),逐组地进行加密。在公钥密码体制中,加密密钥和解密密钥不同,从一个难于推出另一个,可将加密和解密能力分开。现有的大多数公钥密码属于分组密码,只有概率加密体制属于流密码。

在消息传输和处理系统中,除了意定的接收者外,还有非授权者,他们通过各种办法如搭线窃听、电磁窃听、声音窃听等来窃取机密信息,称其为截收者(eavesdropper)。他们虽然不知道系统所用的密钥,但通过分析可能从截获的密文推断出原来的明文,这一过程称作密码分析(cryptanalysis)。从事这一工作的人称作密码分析员或密码分析者(cryptanalyst)。对一个密码系统采取截获密文进行分析的这类攻击称作被动攻击(passive attack)。密码系统还可能遭受的另一类攻击是主动攻击(active attack),非法入侵者(tamper)主动向系统窜扰,采用删除、更改、增填、重放、伪造等手段向系统注入假消息,以达到损人利己的目的。所谓一个密码是可破的(breakable),是指如果通过密文能够迅速地确定明文或密钥,或通过明文-密文对能够迅速地确定密钥。通常假定密码分析者或敌手(opponent)知道所使用的密码系统,这个假设称作 Kerckhoff 假设。当然,如果密码分析者或敌手不知道所使用的密码系统,那么破译密码是更难的,但是我们不应该把密码系统的安全性建立在敌手不知道所使用的密码系统这个前提之下。因此,在设计一个密码系统时,我们的目的是在 Kerckhoff 假设下达到安全性。

破译或攻击(break 或 attack)密码的方法有穷举破译法(exhaustive attack method)和分析法两种。穷举法又称作强力法(brute force method)或完全试凑法(complete trial-and-error method),这种方法是对截获的密文依次用各种可能的密钥试译,直到得到有意义的明文,或在密钥不变的情况下,对所有可能的明文加密直到得到与截获密文一致为止。只要有足够的计算时间和存储空间,原则上穷举法总是可以成功的,但在实际中,时间和存储空间都受到约束,因此,这种方法往往是不可行的。分析破译法又分确定性分析法和统计分析法两类。确定性分析法是利用一个或几个已知量用数学关系式表示出所求未知量(如密钥等)。统计分析法是利用明文的已知统计规律进行破译的方法,密码分析者对截收的密文进行统计分析,总结出其间的统计规律,并与明文的统计规律进行对照比较,从中提取出明文和密文之间的对应或变换信息。密码分析之所以能够成功地破译密码,最根本的原因是明文中有多余度。

根据密码分析者破译时已具备的前提条件,通常人们将攻击类型分为四种,即唯密文攻击(ciphertext-only attack)、已知明文攻击(known plaintext attack)、选择明文攻击(chosen plaintext attack)和选择密文攻击(chosen ciphertext attack)。

(1)唯密文攻击:密码分析者有一个或更多的用同一个密钥加密的密文,通过对这些截获的密文进行分析得出明文或密钥。

(2)已知明文攻击:除待解的密文外,密码分析者有一些明文和用同一个密钥加密这些明文所对应的密文。

(3)选择明文攻击:密码分析者可得到所需要的任何明文所对应的密文,这些密文与待解的密文是用同一个密钥加密得来的。

(4)选择密文攻击:密码分析者可得到所需要的任何密文所对应的明文(这些明文可能是不大明了的),解密这些密文所使用的密钥与解密待解的密文的密钥是一样的。

上述四种攻击类型的强度按序递增,唯密文攻击是最弱的一种攻击,选择密文攻击是最强的一种攻击。选择密文攻击主要用于分析公钥密码体制。如果一个密码系统能够抵抗选择明文攻击,那么它当然能够抵抗唯密文攻击和已知明文攻击。

一个密码通信系统可用图 1.1.1 表示,它由以下几个部分组成:明文消息空间 P;密文消息空间 C;密钥空间 K_1 和 K_2,在私钥体制下 $K_1=K_2=K$,此时密钥 K 需经安全的密钥信道由发方传送给收方;加密变换 $E_{k_1}:P\to C,k_1\in K_1$,由加密器完成;解密变换 $D_{k_2}:C\to P,k_2\in K_2$,由解密器实现。对每一个密钥 $k_1\in K_1$(k_1 确定一个加密变换 E_{k_1}),有一个匹配的密钥 $k_2\in K_2$(k_2 确定一个解密变换 D_{k_2})使得对一切 $m\in P$,有 $D_{k_2}(E_{k_1}(m))=m$。

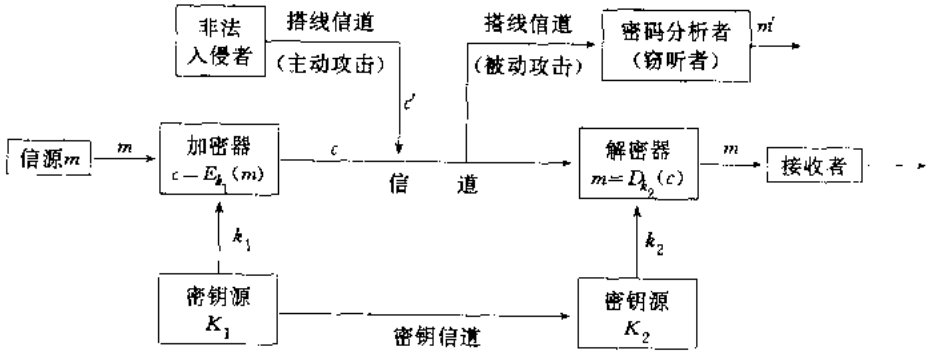


图1.1.1 密码系统模型

对于给定的明文消息 $m\in P$ 和密钥 $k_1\in K_1$,加密变换将明文 m 变换为密文 c:

$$c=f(m,k_1)=E_{k_1}(m),\qquad m\in P,\qquad k_1\in K_1 \tag{1.1.1}$$

接收者利用通过安全信道传送来的密钥 k_1(私钥体制下)或用本地密钥生成器产生的解密密钥 $k_2\in K_2$(公钥体制下)控制解密操作 D,对收到的密文进行变换恢复明文消息 m:

$$m=D_{k_2}(c),m\in P,k_2\in K_2 \tag{1.1.2}$$

而密码分析者,则用选定的变换函数 h,对截获的密文 c 进行变换,得到的明文是明文空间中的某个元素 m'

$$m'=h(c) \tag{1.1.3}$$

一般地,$m'\neq m$。

令 $\varepsilon=\{E_{k_1}:P\to C\mid k_1\in K_1\}$,$D=\{D_{k_2}:C\to P\mid k_2\in K_2\}$,则称六元组$(P,C,K_1,K_2,\varepsilon,D)$为一保密系统(secrecy system)。

为了保护信息的机密性,抵抗密码分析,保密系统应当满足下述要求:

(1)系统即使达不到理论上是不可破的,即 $p_r(m'=m)=0$,也应当是实际上不可破

的。也就是说，从截获的密文或某些已知明文-密文对，要确定密钥或任意明文在计算上是不可行的。

(2)系统的保密性不依赖于对加密体制或算法的保密(Kerckhoff 假设)，而依赖于密钥。

(3)加密和解密算法适用于所有密钥空间中的元素。

(4)系统既易于实现又便于使用。

防止消息被窜改、删除、重放和伪造的一种有效的方法是使发送的消息具有被验证的能力，使接收者或第三者能够识别和确认消息的真伪，实现这类功能的密码系统称作认证系统(authentication system)。消息的认证性和消息的保密性不同，保密性是使截获者在不知道密钥的条件下不能解读密文的内容，而认证性是使任何不知道密钥的人不能构造出一个密报，使意定的接收者解密成一个可理解的消息(合法的消息)。认证理论和技术是最近 20 年来随着计算机通信的普遍应用而迅速发展起来的一个重要的密码学研究领域。如传统的手写签名正在被更迅速、更经济和更安全的数字签名(digital signature)所取代。

一个安全的认证系统应当满足下述的基本要求：

(1)意定的接收者能够检验和证实消息的合法性和真实性。

(2)消息的发送者对所发送的消息不能抵赖。

(3)除了合法的消息发送者外，其他人不能伪造合法的消息。而且在已知合法密文 c 和相应消息 m 下，要确定加密密钥或系统地伪造合法密文在计算上是不可行的。

(4)当通信双方(或多方)发生争执时，可由称作仲裁者(arbitrator)的第三方解决争执。

这里值得一提的是，密码学中的术语“系统或体制”(system)、“方案”(scheme)和“算法”(algorithm)本质上是一回事，本书中按作者的习惯交替使用了这些术语。

1.2 古典密码学

本节简要介绍几种古典密码体制及对这些体制的一些破译方法，用来说明设计和分析密码的基本方法。虽然这些密码大都比较简单而且容易破译，但研究这些密码的设计原理和分析方法对于理解、设计和分析现代密码是十分有益的。

1.2.1 古典密码体制

1.2.1.1 代换密码和置换密码

令 A 表示含 N 个“字母”或“字符”的明文字母表，例如，可以是普通的英文字母 $A\sim Z$，也可以是数字、空格、标点符号或任何可以表示明文消息的符号。因此可以将 A 抽象地表示为一个整数集 $Z_N=\{0,1,\cdots,N-1\}$。在加密时通常将明文消息划分成长为 L 的消息单元，称为明文组，以 m 表示，如 $m=(m_0,m_1,\cdots,m_{L-1})$，$m_i\in Z_N$，$0\leqslant i\leqslant L-1$。$m$ 也称作 L-报文，它是定义在 Z_N^L 上的随机变量，$Z_N^L=Z_N\times Z_N\times\cdots\times Z_N$($L$ 个)$=\{m=(m_0,m_1,\cdots,m_{L-1})\mid m_i\in Z_N,0\leqslant i\leqslant L-1\}$。$L=1$ 为单字母报(1-gram)，$L=2$ 为双字母报(digrams)，$L=3$ 为三字母报(trigrams)。明文空间 $P=Z_N^L$。

令 A' 表示含 N' 个“字母”或“字符”的密文字母表，抽象地可用整数集 $Z_{N'}=\{0,1,\cdots,N'-1\}$ 来表示。密文单元或组为 $c=(c_0,c_1,\cdots,c_{L'-1})$（$L'$ 个），$c_{l'}\in Z_{N'}$，$0\leqslant l'\leqslant L'-1$。$c$ 是定义在 $Z_{N'}^{L'}$ 上的随机变量。密文空间 $C=Z_{N'}^{L'}$。

一般地，明文和密文由同一字母表构成，即 $A'=A$。

加密变换是从明文空间到密文空间的映射 $f:P\to C$。加密变换通常是在密钥控制下变化的，因此，一般记为

$$c=f(m,k)=E_k(m)\qquad k\in K\qquad m\in P\qquad c\in C \tag{1.2.1}$$

K 为密钥空间。

假定 f 是一个单射，对固定的 $k\in K$，令 $C_k=\{c=f(m,k)=E_k(m)\mid m\in P\}\subseteq C$，因此对给定的密文组 $c\in C_k$，有且仅有一个对应的明文组 m，也就是说，对于此函数 f，存在逆映射 $f^{-1}:C_k\to P$，使

$$f^{-1}(c)=f^{-1}\{f(m)\}=m\qquad m\in P\qquad c\in C_k \tag{1.2.2}$$

即 f^{-1} 为解密变换。

一个密码系统就是在 f 作用下由 Z_N^L 到 $Z_{N'}^{L'}$ 的映射，在这种意义上，称此种密码为代换密码(substitution cipher)，如图 1.2.1 所示。$L=1$ 时，称作单字母代换，也称作流密码(stream cipher)。$L>1$ 时，称作多码代换，也称作分组密码(block cipher)。

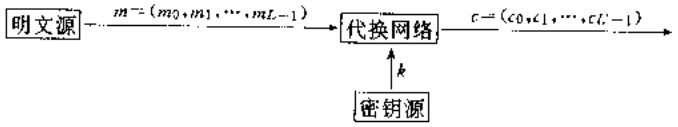


图 1.2.1　代换密码框图

一般地，选择相同的明文和密文字母表。此时，若 $L=L'$，则映射 f 可构造成一对一的映射，密码无数据扩展。若 $L<L'$，则有数据扩展，可将映射 f 设计成一对多的映射，即明文组可能找到多于一个密文组来代换，这称之为多名(或同音)代换密码(homophonic substitution cipher)。若 $L>L'$，则明文数据将被压缩，此时映射 f 不可能构造成可逆映射，从而从密文有时也就无法完全恢复出原明文消息，因此保密通信中必须要求 $L\leqslant L'$。但 $L>L'$ 的映射可以用在认证系统中。

在 $A=A'$，$N=N'$ 和 $L=1$ 时，若对所有明文字母，都用一种固定的代换进行加密，则称这种密码为单表代换(monoalphabetic substitute)。若用一个以上的代换表进行加密，这就称作是多表代换(polyalphabetic substitute)。这是古典密码中的两种重要体制，曾被广泛地使用过。

在代换密码中有一种特殊的代换密码，即代换并没有改变明文字母，而只改变了它们的位置，密码学史上把这种代换密码称作置换密码(permutation cipher)，又称换位密码(transposition cipher)。下面是置换密码的一个详细描述。

设 $P=C=Z_N^L$，K 表示所有的 $\{0,1,\cdots,L-1\}$ 上的置换构成的集合。对每一个给定的密钥 $k=\pi$（一个置换），置换密码的加密变换定义为 $E_\pi(m_0,m_1,\cdots,m_{L-1})=(m_{\pi(0)},m_{\pi(1)},\cdots,m_{\pi(L-1)})=(c_0,c_1,\cdots,c_{L-1})$。因而解密变换为 $D_{\pi^{-1}}(c_0,c_1,c_{L-1})=(c_{\pi^{-1}(0)},c_{\pi^{-1}(1)},\cdots,c_{\pi^{-1}(L-1)})$，这里 π^{-1} 是 π 的逆置换。

1.2.1.2 单表代换密码

单表代换密码是对明文的所有字母都用一个固定的明文字母表到密文字母表的映射，即 $f:Z_N \to Z_N$。令明文 $m=m_0m_1\cdots$，则相应的密文为 $c=E(m)=c_0c_1\cdots=f(m_0)f(m_1)\cdots$。若明文字母表为 $A=Z_N=\{0,1,\cdots,N-1\}$，则相应的明文字母表为 $A'=\{f(0),f(1),\cdots,f(N-1)\}$，$A'$ 是 A 的某种置换。本小节主要来介绍一类简单的单表代换密码——仿射密码。

仿射密码(affine cipher)是一种特殊的代换密码。明文空间 P 和密文空间 C 均为 Z_N，密钥空间为 $K=\{k=(k_1,k_0)\mid \gcd(k_1,N)=1,k_0,k_1\in Z_N\}$。对给定的密钥 $k=(k_1,k_0)\in K$，其加密变换为

$$E_k(i)=(ik_1+k_0)\bmod N \tag{1.2.3}$$

解密变换为

$$D_k(j)=k_1^{-1}(j-k_0)\bmod N \tag{1.2.4}$$

其中 $\gcd(k_1,N)$ 表示 k_1 与 N 的最大公因子，$x \bmod N$ 表示 x 除以 N 所得的余数，密钥空间 K 的大小为 $N\varphi(N)$。关于剩余类环 Z_N 和欧拉函数 $\varphi(\cdot)$的定义和性质参见附录。

当 $k_0=0$ 时，称为乘法密码(multiplicative cipher)，又称采样密码(decimation cipher)。当 $k_0=1$ 时，称为移位密码(shift cipher)，又称加法密码(additive cipher)。

我们将通过建立英文字母和模 26 的剩余之间的对应关系来使用移位密码加密普通的英文消息。

例 1.2.1 假定移位密码的密钥为 $k_0=3$，明文消息为 we will meet at midnight。我们首先利用表 1.2.1 将该明文转化成一列整数：22 4 22 8 11 11 12 4 4 19 0 19 12 8 3 13 8 6 7 19。其次加 3 到每个值，并进行模 26 运算得：7 15 7 19 22 22 23 15 15 4 11 4 23 19 14 24 19 17 18 4。最后，再利用表 1.2.1 将这一列整数转化为字母，从而获得密文：HPHTWWXPPELEXTOYTRSE。解密过程与加密过程类似，不同的只是进行模 26 减 3，而不是模 26 加 3。

表 1.2.1 英文字母和模 26 的剩余之间的对应关系

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

1.2.1.3 多表代换密码

多表代换密码是以一系列(两个以上)代换表依次对明文消息的字母进行代换的加密方法。令明文字母表为 Z_N，$f=(f_1,f_2,\cdots)$为代换序列，明文字母序列 $m=m_1m_2\cdots$，则相应的密文字母序列为 $c=E_k(m)=f(m)=f_1(m_1)f_2(m_2)\cdots$。若 f 是非周期的无限序列，则相应的密码称为非周期多表代换密码。这类密码，对每个明文字母都采用不同的代换表(或密钥)进行加密，称作一次一密密码(one-time pad cipher)，这是一种在理论上唯一不可破的密码(参见第 2 章)。这种密码可以完全隐蔽明文的特点，但由于需要的密钥量和明文消息长度相同而难于广泛使用。为了减少密钥量，在实际应用中多采用周期多表代换密码，即代换表个数有限，重复地使用，此时代换表序列为 $f=(f_1,f_2,\cdots,f_d,f_1,f_2,\cdots,f_d,\cdots)$，

相应于明文字母 m 的密文为 $c=E_k(m)=f(m)=f_1(m_1)f_2(m_2)\cdots f_d(m_d)f_1(m_{d+1})f_2(m_{d+2})\cdots f_d(m_{2d})\cdots$。当 $d=1$ 时就退化为单表代换，因此可以说多表代换密码是单表代换密码的一种推广。

有名的多表代换密码有 Vigenère、Beaufort、Running-Key、Vernam 和转轮机(rotor machine)等密码。这里只介绍 Vigenère 多表代换密码，其它的参见文献[7,8,9,10]等。

Vigenère 密码是由法国密码学家 Blaise de Vigenère 于 1858 年提出的一种密码，它是一种以移位代换(当然也可以用一般的字母代换表)为基础的周期代换密码。d 个代换表 $f=(f_1,f_2,\cdots,f_d)$ 由 d 个字母序列给定的密钥 $k=(k_1,k_2,\cdots,k_d)\in Z_N^d$ 决定，其中 $k_i(i=1,2,\cdots,d)$ 确定明文的第 $i+td$ 个字母(t 为正整数)的移位次数，即加密公式为

$$c_{i+td}=E_{k_i}(m_{i+td})=(m_{i+td}+k_i)\bmod N \tag{1.2.5}$$

从而解密公式为

$$\begin{aligned} m_{i+td}&=D_{k_i}(c_{i+td})=E_{N-k_i}(c_{i+td})\\ &=(N-k_i+m_{i+td}+k_i)\bmod N=m_{i+td} \end{aligned} \tag{1.2.6}$$

称 k 为用户密钥(user key)或密钥字(key word)。密钥量为 N^d，当 N 与 d 较大时，密钥量是很大的。将用户密钥 k 周期地延伸就给出了整个明文加密所需的工作密钥(working key)。

例 1.2.2 假定我们仍使用表 1.2.1，$d=6$，$k=$cipher，明文串是 this cryptosystem is not secure。

首先将 k 及明文串转化为数字串：$k=(2,8,15,7,4,17)$，$m=(19,7,8,18,2,17,24,15,19,14,18,24,18,19,4,12,8,18,13,14,19,18,4,2,20,17,4)$。

其次模 26“加”密钥字 $k=(2,8,15,7,4,17)$得：

19	7	8	18	2	17		24	15	19	14	18	24
2	8	15	7	4	17		2	8	15	7	4	17
21	15	23	25	6	8		0	23	8	21	22	15
18	19	4	12	8	18		13	14	19	18	4	2
2	8	15	7	4	17		2	8	15	7	4	17
21	1	19	19	12	9		15	22	8	25	8	19
20	17	4										
2	8	15										
22	25	19										

最后将所得的密文数字串利用表 1.2.1 转化成密文字母串即

VPXZGIAXIVWPUBTTMJPWIZITWZT

解密过程与加密过程类似，不同的只是进行模 26 减，而不是模 26 加。

1.2.1.4 多字母代换密码

前面介绍的仿射密码和 Vigenère 密码都是以单个字母作为代换对象的。如果每次对 $L>1$ 个字母进行代换就是多字母代换密码(polygram substitution cipher)。多字母代换的优点是容易将字母的自然频度隐蔽或均匀化而有利于抵抗统计分析。这种密码主要有 Playfair 密码、Hill 密码等。这里主要介绍 Hill 密码,其它的参见文献[7,8,9,10]等。

令明文字母表为 Z_N,若采用 L 个字母为单元进行代换,则多码代换是映射 $f:Z_N^L \to Z_N^L$。若映射 f 是线性的,则称 f 是线性变换,可用一个 Z_N 上的 $L\times L$ 阶矩阵 T 表示。若 T 是满秩的,则变换为一对一映射,且存在逆变换 T^{-1},使 $TT^{-1}=T^{-1}T=I$(I 为 $L\times L$ 阶单位矩阵)。将 L 个字母的数字表示为 Z_N 上的一个向量 $m=(m_1,m_2,\cdots,m_L)$,则 Hill 密码的加密变换为

$$c=(c_1,c_2,\cdots,c_L)=mT \bmod N \tag{1.2.7}$$

解密变换为

$$m=cT^{-1} \bmod N \tag{1.2.8}$$

类似于单字母仿射代换密码,可构造多字母仿射代换密码。令 $b=(b_1,b_2,\cdots,b_L)\in Z_N^L$,$T$ 为 Z_N 上的 $L\times L$ 阶满秩矩阵,则可通过下述仿射变换对明文组 $m=(m_1,m_2,\cdots,m_L)$加密得密文 $c=(c_1,\cdots,c_L)$,即

$$c=(mT+b) \bmod N \tag{1.2.9}$$

解密变换为

$$m=(c-b)T^{-1} \bmod N \tag{1.2.10}$$

当 $T=I$ 时,就是上一小节介绍的 Vigenère 密码。

例 1.2.3 假定 $L=2$,明文空间和密文空间均为 $P=C=Z_{26}$,密钥

$$K=\begin{pmatrix}11 & 8\\ 3 & 7\end{pmatrix},\ K^{-1}=\begin{pmatrix}7 & 18\\ 23 & 11\end{pmatrix}$$

现在加密明文 july,由表 1.2.1 知,ju 对应于数字组(9,20),ly 对应于数字组(11,24)。计算$(9,20)K \bmod 26=(3,4)$,$(11,24)K \bmod 26=(11,22)$。再由表 1.2.1 知,(3,4)对应于 DE,(11,22)对应于 LW,故密文为 DELW。

解密过程:由表 1.2.1 知,DE 对应于(3,4),LW 对应于(11,22)。计算$(3,4)K^{-1} \bmod 26=(9,20)$,$(11,22)K^{-1} \bmod 26=(11,24)$。再由表 1.2.1 知,(9,20)对应于 ju,(11,24)对应于 ly,故明文为 july。

1.2.2 古典密码体制分析

简单的单表代换密码,如移位密码极易破译。仅统计标出最高频度字母再与明文字母表字母对应决定出移位量,就差不多可以得到正确解了。其它如乘法密码、一般的仿射密码要复杂些,但多考虑几个密文字母统计表与明文字母统计表的匹配关系也不难解出。另外单表代换密码如移位密码也很容易用穷举密钥搜索来破译,因为密钥量仅为 N。可见,一个密码系统是安全的一个必要条件是密钥空间必须足够的大,使得穷举密钥搜索破译是不可行的,但这不是一个密码系统安全的充分条件。

多表代换密码的破译要比单表代换密码的破译难得多,因为在单表代换下,字母的频

度、重复字母模式、字母结合方式等统计特性除了字母名称改变以外，都未发生变化，依靠这些不变的统计特性就能破译单表代换，而在多表代换下，原来明文中的这些特性通过多个表的平均作用而被隐蔽了起来。已有事实表明，用唯密文攻击法分析单表和多表代换密码是可行的，但用唯密文攻击法分析多字母代换密码如 Hill 密码是比较困难的。分析多字母代换密码多用已知明文攻击法。本节我们以 Vigenère 密码为例来说明多表代换密码的一些分析方法，其它古典密码的分析参见文献[7,8,,10,11,12,13]等。

分析 Vigenère 密码的第一步是确定密钥字的长度 d。确定密钥字的长度 d 的方法最常用的有两种即 Kasiski 测试法和重合指数(index of coincidence)法。

Kasiski 测试法是由普鲁士军官 Kasiski 在 1863 年提出的一种重码分析法。这种方法的基本原理是：若用给定的 d 个密钥表周期地对明文字母加密，则当明文中有两个相同字母组在明文序列中间隔的字母数为 d 的倍数时，这两个明文字母组对应的密文字母组必相同。但反过来，若密文中出现两个相同的字母组，它们所对应的明文字母组未必相同，但相同的可能性很大。如果我们将密文中相同字母组找出来，并对其相间字母数综合研究，找出它们的相同字母数的最大公因子，就有可能提取出有关密钥字的长度 d 的信息。

下面我们来介绍确定 Vigenère 密码的密钥字的长度 d 的另一种方法——重合指数法。

定义 1.2.1 设 $x=x_1x_2\cdots x_n$ 是 n 个字母的一个串，x 的重合指数定义为 x 中的两个随机元素相同的概率，记为 $I_c(x)$。

假定 $f_0,f_1,\cdots,f_{25}$ 分别表示 x 中字母 $A,B,C,\cdots,Z$ 出现的频率。我们能以 $\binom{n}{2}$ 种方法选择 x 中的两个元素，对每一个 i，$0\leqslant i\leqslant 25$，x 中的两个元素都被选择为 i 的方法有 $\binom{f_i}{2}$ 种，因此

$$I_c(x)=\frac{\sum_{i=0}^{25}\binom{f_i}{2}}{\binom{n}{2}}=\frac{\sum_{i=0}^{25}f_i(f_i-1)}{n(n-1)} \tag{1.2.1}$$

现在假定 x 是一个英文明文串。记字母 A,B,…,Z 出现的期望概率分别为 $p_0,p_1,\cdots,p_{25}$。通过对大量的小说、杂志、报纸等的汇编统计，人们已经获得了英文的 26 个字母的概率分布的一个估计，见表 1.2.2。

表 1.2.2 26 个英文字母出现的概率

字母	概率	字母	概率
A	0.082	N	0.067
B	0.015	O	0.075
C	0.028	P	0.019
D	0.043	Q	0.001
E	0.127	R	0.060
F	0.022	S	0.063
G	0.020	T	0.091
H	0.061	U	0.028
I	0.070	V	0.010
J	0.002	W	0.023
K	0.008	X	0.001
L	0.040	Y	0.020
M	0.024	Z	0.001

我们将期望 $I_c(x)\approx\sum_{i=0}^{25}p_i^2=0.065$，这是因为两个随机元素都是 A 的概率为 p_0^2，两个随机元素都是 B 的概率为 p_1^2，等等。如果 x 是利用任何多表代换密码获得的一个密文，那么在这种情况下，各个概率将只是被作了一个置换，但量 $\sum_{i=0}^{25}p_i^2$ 是不变的。因此，对用多表代换密码获得的一个密文 x，也

有 $I_c(x)\approx\sum_{i=0}^{25}p_i^2=0.065$。

假定 $y=y_1y_2\cdots y_n$ 是通过 Vigenère 密码所获得的一个密文串。把 y 分成 d 个长为 n/d 的子串，记为 $Y_1,Y_2,\cdots,Y_d$。如果 d 的确是密钥字长度，那么每个 $I_c(Y_i)(1\leqslant i\leqslant d)$ 都将大概等于 0.065。如果 d 不是密钥字的长度，那么子串 Y_i 将看起来更随机些，因为它们将是采用不同的密钥移位加密获得的。而对一个完全随机的串 x，$I_c(x)\approx 26\left(\frac{1}{26}\right)^2=\frac{1}{26}=0.038$。因为两个值 0.065 和 0.038 间隔的充分远，所以我们通常能确定正确的密钥字的长度。

例 1.2.4 假定有一段来自用 Vigenère 密码加密的密文：

CHREEVOAHMAERATBIAXXWTNXBEEOPHBSBQMQEQERBW
RVXUOAKXAOSXXWEAHBWGJMMQMNKGRFVGXWTRZXWI
AKLXFPSKAUTEMNDCMGTSXMXBTUIADNGMGPSRELXNJEL
XVRVPRTULHDNQWTWDTYGBPHXTFALJHASVBFXNGLLCHR
ZBWELEKMSJIKNBHWRJGNMGJSGLXFEYPHAGNRBIEQJTAM
RVLCRREMNDGLXRRIMGNSNRWCHRQHAEYEVTAQEBBIPEE
WEVKAKOEWADREMXMTBHHCHRTKDNVRZCHRCLQOHPWQ
AIIWXNRMGWOIIFKEE

首先我们用 Kasiski 测试法确定密钥字的长度。密文串 CHR 在密文中的四处出现，起始位置在 1，166，236 和 286。从第 1 个到其它 3 个的距离分别为 165，235 和 285，这三个数的最大公因子是 5，所以 5 很可能是密钥字的长度。

其次我们计算一下重合指数，看是否与 Kasiski 测试法所得的结果一致。当 $d=1$ 时，重合指数为 0.045；当 $d=2$ 时，两个重合指数分别为 0.046 和 0.041；当 $d=3$ 时，三个重合指数分别为 0.043，0.050 和 0.047；当 $d=4$ 时，四个重合指数分别为 0.042，0.039，0.046 和 0.040；当 $d=5$ 时，五个重合指数分别为 0.063，0.068，0.069，0.061 和 0.072，这个也为密钥字的长度是 5 提供了有力的证据。

分析 Vigenère 密码的第二步是确定密钥字(密钥字的长度已由第一步确定)。通常采用重合互指数(mutual index of coincidence)法来确定密钥字，重合互指数的精确定义如下。

定义 1.2.2 假定 $x=x_1x_2\cdots x_n$ 和 $y=y_1y_2\cdots y_{n'}$ 分别是长为 n 和 n' 的字母串。x 和 y 的重合互指数定义为 x 的一个随机元素等于 y 的一个随机元素的概率，记为 $MI_c(x,y)$。

如果我们将 x 和 y 中的字母 A，B，C，…，Z 出现的频率分别表示为 $f_0,f_1,\cdots,f_{25}$ 和 $f_0,f_1,\cdots,f_{25}$，那么

$$\mathrm{MI}_c(x,y)=\sum_{i=0}^{25}\frac{f_i}{n}\cdot\frac{f_i}{n'}=\frac{\sum_{i=0}^{25}f_if_i}{nn'} \tag{1.2.12}$$

现在假定我们已经确定了 d 的值，子串 Y_i 是通过明文的移位加密获得的，并假定 $K=(k_1,k_2,\cdots,k_d)$ 是密钥字。我们来估计 $\mathrm{MI}_c(Y_i,Y_j)$。考虑 Y_i 中的一个随机字母和 Y_j 中的一个随机字母，两个字母都是 A 的概率是 $p_{-k_i}p_{-k_j}$，两个字母都是 B 的概率是 $p_{1-k_i}p_{1-k_j}$，等等(注意，所有的下标都是模 26 运算)。因此，我们可估计

$$\mathrm{MI}_c(Y_i,Y_j)\approx\sum_{h=0}^{25}p_{h-k_i}p_{h-k_j}=\sum_{h=0}^{25}p_hp_{h+k_i-k_j} \tag{1.2.13}$$

易知，$\mathrm{MI}_c(Y_i,Y_j)$的估计值只依赖于差$(k_i-k_j)\mathrm{mon}26$，我们将这个差称为 Y_i 和 Y_j 的相对移位(relative shift)。又 $\sum_{h=0}^{25}p_hp_{h+l}=\sum_{h=0}^{25}p_hp_{h-l}$，所以 MI_c 关于相对移位 l 和 $26-l$ 的估计值是一样的。

由表 1.2.3 知，当相对移位不是 0 时，这些估计值均在 0.031～0.045 之间，而相对移位是 0 时，估计值是 0.065。我们能使用这个表来推测 Y_i 和 Y_j 的相对移位 $l=(k_i-k_j)\mathrm{mod}26$。具体做法如下：假定我们固定 Y_i，考虑由长为 d 的密钥字 $e_0=(0,0,\cdots,0)$，$e_1=(1,1,\cdots,1)$，$e_2=(2,2,\cdots2)$，…加密 Y_j 的影响，将加密结果分别记为 $Y_j^0,Y_j^1,Y_j^2,\cdots$。利用公式 $\mathrm{MI}_c(x,y^g)=\dfrac{\sum_{i=0}^{25}f_if_{i-g}}{nn'}$ 容易计算 $\mathrm{MI}_c(Y_i,Y_j^g)$，$0\leqslant g\leqslant25$。当 $g=l$ 时，MI_c 将接近 0.065，因为 Y_i 和 Y_j^l 的相对移位是 0。然而对 $g\neq l$ 的值，MI_c 的值将在 0.031 和 0.045 之间。通过使用重合互指数这种技术，我们能获得子串 $Y_1,Y_2,\cdots$中的任何两个的相对移位。

表 1.2.3　期望的重合指数

相对移位	MI_c 的期望值
0	0.065
1(25)	0.039
2(24)	0.032
3(23)	0.034
4(22)	0.044
5(21)	0.033
6(20)	0.036
7(19)	0.039
8(18)	0.034
9(17)	0.034
10(16)	0.038
11(15)	0.045
12(14)	0.039
13	0.043

例 1.2.4(续)　我们已经假定了密钥字的长度是 5。现在极力计算相对移位。利用计算机不难算出 260 个值 $\mathrm{MI}_c(Y_i,Y_j^g)$，$1\leqslant i<j\leqslant5$，$0\leqslant g\leqslant25$。这些值见表 1.2.4。

表 1.2.4

i	j	$\mathrm{MI}_c(Y_i,Y_j^g)$的值								
1	2	0.028	0.027	0.028	0.034	0.039	0.037	0.026	0.025	0.052
		0.068	0.044	0.026	0.037	0.043	0.037	0.043	0.037	0.028
		0.041	0.041	0.034	0.037	0.051	0.045	0.042	0.036	
1	3	0.039	0.033	0.040	0.034	0.028	0.053	0.048	0.033	0.029
		0.056	0.050	0.045	0.039	0.040	0.036	0.037	0.032	0.027
		0.037	0.036	0.031	0.037	0.055	0.029	0.024	0.037	
1	4	0.034	0.043	0.025	0.027	0.038	0.049	0.040	0.032	0.029
		0.034	0.039	0.044	0.044	0.034	0.039	0.045	0.044	0.037
		0.055	0.047	0.032	0.027	0.039	0.037	0.039	0.035	
1	5	0.043	0.033	0.028	0.046	0.043	0.044	0.039	0.031	0.026
		0.030	0.036	0.040	0.041	0.024	0.019	0.048	0.070	0.044
		0.028	0.038	0.044	0.043	0.047	0.033	0.026	0.046	
2	3	0.046	0.048	0.041	0.032	0.036	0.035	0.036	0.030	0.024
		0.039	0.034	0.029	0.040	0.067	0.041	0.033	0.037	0.045
		0.033	0.033	0.027	0.033	0.045	0.052	0.042	0.030	
2	4	0.046	0.034	0.043	0.044	0.034	0.031	0.040	0.045	0.040
		0.048	0.044	0.033	0.024	0.028	0.042	0.039	0.026	0.034
		0.050	0.035	0.032	0.040	0.056	0.043	0.028	0.028	
2	5	0.033	0.033	0.036	0.046	0.026	0.018	0.043	0.080	0.050

续表 1.2.4

i	j	$MI_c(Y_i,Y_j^g)$的值								
		0.029	0.031	0.045	0.039	0.037	0.027	0.026	0.031	0.039
		0.040	0.037	0.041	0.046	0.045	0.043	0.035	0.030	
3	4	0.038	0.036	0.040	0.033	0.036	0.060	0.035	0.041	0.029
		0.058	0.035	0.035	0.034	0.053	0.030	0.032	0.035	0.036
		0.036	0.028	0.046	0.032	0.051	0.032	0.034	0.030	
3	5	0.035	0.034	0.034	0.036	0.030	0.043	0.043	0.050	0.025
		0.041	0.051	0.050	0.035	0.032	0.033	0.033	0.052	0.031
		0.027	0.030	0.072	0.035	0.034	0.032	0.043	0.027	
4	5	0.052	0.038	0.033	0.038	0.041	0.043	0.037	0.048	0.028
		0.028	0.036	0.061	0.033	0.033	0.032	0.052	0.034	0.027
		0.039	0.043	0.033	0.027	0.030	0.039	0.048	0.035	

对每对(i,j)，找出接近于 0.065 的$MI_c(Y_i,Y_j^g)$的值。如果对一个给定的对(i,j)有唯一的一个这样的值，我们就把它视作相对移位的值(也可能不是)。在表 1.2.4 中，用方框已划出了六个这样的值，它们提供了如下的信息：Y_1 和 Y_2 的相对移位可能是 9；Y_1 和 Y_5 的相对移位可能是 16；Y_2 和 Y_3 的相对移位可能是 13；Y_2 和 Y_5 的相对移位可能是 7；Y_3 和 Y_5 的相对移位可能是 20；Y_4 和 Y_5 的相对移位可能是 11。这些关系给出了关于未知量 k_1,k_2,k_3,k_4,k_5 的如下关系式：

$$k_1-k_2=9 \quad k_1-k_5=16 \quad k_2-k_3=13 \quad k_2-k_5=7 \quad k_3-k_5=20 \quad k_4-k_5=11$$

这样 $k_2=k_1+17,k_3=k_1+4,k_4=k_1+21,k_5=k_1+10$。所以密钥字的可能形式为$(k_1,k_1+17,k_1+4,k_1+21,k_1+10),k_1\in Z_{26}$。不难确定密钥字是 JANET(最多穷搜索 26 种可能)。

将密文可解密为：The almond tree was in tentative blossom. The days were longer, often ending with magnificent evenings of corrugated pink skies. The hunting season was over, with hounds and guns put away for six months. The vineyards were busy again as the well-organized farmers treated their vines and the more lackadaisical neighbors hurried to do the pruning they should have done in November.

1.3 注记和文献

本章主要介绍了密码学的一些基本概念和一些有代表性的古典密码体制，并对 Vigenère 密码作了详细分析。古典密码体制的分析方法如 Vigenère 密码的分析方法至今仍具有重要的参考价值。本章主要参考了文献[10,14]。

关于密码学发展和应用方面的综述论文有很多，可参阅文献[8,15,16,17,18,19,20]等。关于密码学发展史方面的论文可参阅文献[1]和 Cryptologia 杂志等。

关于密码学的教材或专著，国内外已出版了许多本，如文献[7,8,9,10,14,21,22,23,24,25,26,27,28,29,30]等。

关于密码学的会议文集有 Asiacrypt 会议文集(已出版的有 Asiacrypt'91,

Asiacrypt'94，Asiacrypt'96，Asiacrypt'98），Auscrypt 会议文集（已出版的有 Auscrypt'90，Auscrypt'92，），Crypto 会议文集（已出版的有 Crypto'81－Crypto'98），Eurocrypt 会议文集（已出版的有 Eurocrypt'82，Eurocrypt'84－Eurocrypt'98），快速软件加密会议文集，IEEE 安全与保密研讨会文集，Chinacrypt 会议文集等等。

关于密码学的杂志有 Journal of Cryptology，Cryptologia，Computer & Security，Designs，Code and Cryptography，《通信保密》，《密码与信息》等。当然在与计算机、通信、数学、信息论等学科有关的杂志上也经常刊登一些密码学方面的论文。

关于密码学的参考文献可进一步参阅文献[10，18，28，29]中所列的参考文献，那几本书中的参考文献比较全。

参 考 文 献

[1] Kahn，D.，The Codebreakers：The Story of Secret Writing，New York：Macmillan，1967.

[2] Feistel，H.，Cryptography and Computer Privacy，Scientific American，Vol. 228，No. 5，May 1973，pp. 15—23.

[3] Smith，J. L.，Notz，W. A. and Osseck，P. R.，An Experimental Application of Cryptography to a Remotely Accessed Data System，Proceedings of the ACM Annual Conference，Aug. 1972，pp. 282—290.

[4] Smith，J. L.，The Design of Lucifer，A Cryptographic Device for Data Communications，IBM Research Report RC 3326，1971.

[5] Diffie，W. and Hellman，M. E.，New Directions in Cryptography，IEEE Trans. Informat. Theory，Vol. IT 22，pp. 644—654，Nov. 1976.

[6] Simmons，G. J.，Symmetric and Asymmetric Encryption Computing Surveys，Vol. 11，No. 4，Dec. 1979，pp. 305—330.

[7] Denning，D. E. R.，Cryptography and Data Security，Addison-Wesley，1982.

[8] Beker，H. J. and Piper，F. C.，Cipher System-The Protection of Communications，Northwood Books，London，1982.

[9] Konheim，A. G.，Cryptography：A primer，John Wiley & Sons，New York，1981.

[10] 王育民、何大可，保密学-基础与应用，西安电子科技大学出版社，西安，1990.

[11] Sinkov，A.，Elementary Cryptanalsis：A Mathematical Approach，1966.

[12] Deavours，C. A. and Kruh，L.，Machine Cryptography and Modern Cryptanalsis，Artech Book，1985.

[13] Kullback，S.，Statistical Method in Cryptanalysis，Acgean Park Press，1976.

[14] Stinson，D. R.，Cryptography：Theory and Practice，CRC Press，1995.

[15] Feistel，H.，Hotz，W. A. and Smith，J. L.，Some Cryptographic Techniques for machine-to machine Data Communications，Proc. IEEE，Vol. 63，pp. 1545－1554，1975.

[16] Diffie，W. and Hellman，M. E.，Privacy and Authentication An Introduction to Cryptography，Proc. IEEE，Vol. 67，pp. 397—426，1979.

[17] Davida，G.，et al.，Cryptography，Symposium on Security and Privacy，pp. 151—161，IEEE Computer Society，1981.

[18] Simmons，G. J. (edited by Simmons，G. J.)，Contemporary Cryptology，IEEE Press，1991.

[19] 肖国镇、卿斯汉，密码学的现状与展望，电子学报，Vol. 15，No. 5，85—96 页，1987.

[20] Massey，J. L.，An Introduction to Contemporary Cryptology，Proc. IEEE，Vol. 76，No. 5，May 1998，pp. 533—549.

[21] Meyer，C. H. and Matyas，S. M.，Cryptography：A New Dimension in Computer Data Security-A Guide for the Design and Implementation of Secure Systems，John Wiley & Sons，1982.

[22] Kranakis，E.，Primality and Cryptography，John Wiley & Sons，1986.

[23] Rueppel，R.，Analysis and Design of Stream Ciphers，Springer-Verlag，1986.

[24] Siegenthaler，T.，Methoden Für den Entwurf von Stream Cipher Systemen，ADAG Administration & Druch AG，

Zurich, 1986.
[25] 丁存生、肖国镇，流密码学及其应用，国防工业出版社，北京，1994.
[26] Ding, C., Xiao, G. and Shan, W., The Stability Theory of Stream Ciphers, Springer-Verlag, New York, 1991.
[27] 杨义先、林须端，编码密码学，人民邮电出版社，北京，1992.
[28] Schneier, B., Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons, 1994.
[29] Menezes, A. J., Van Oorschot, P. and Vanstone, S., Handbook of Applied Cryptography, CRC Press, 1996.
[30] 卢开澄，计算机密码学，清华大学出版社，北京，1990.

第 2 章　密码学的信息理论基础

密码学的信息理论主要包括 Shannon 的保密系统的信息理论和 Simmons 的认证系统的信息理论，本章主要介绍了这两个方面的内容。

2.1　Shannon 的保密系统的信息理论

1949 年，Claude Shannon 发表了一篇题为“保密系统的信息理论”的论文[1]，该论文用信息论的观点对信息保密问题作了全面的阐述，这使信息论成为研究密码学的一个重要理论基础，宣告了科学的私钥密码学时代的到来。

本节将介绍 Shannon 的保密系统的信息理论的基本观点及其在密码中的应用。

2.1.1　保密系统的数学模型

Shannon 从概率统计观点出发研究信息的传输和保密问题[1,2]，将通信系统归纳为图 2.1.1 的框图，将保密系统归纳为图 2.1.2 的框图。通信系统设计的目的是在信道有干扰的情况下，使接收的信息无错或差错尽可能地小。保密系统设计的目的是使窃听者即使在完全准确地收到了接收信号的情况下也无法恢复出原始消息。

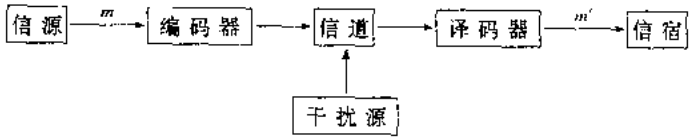


图 2.1.1　通信系统

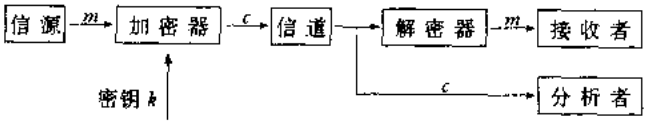


图 2.1.2　保密系统

与通信系统相类似，对保密系统的各部分可作如下描述。

信源是产生消息的源，在离散情况下可以产生字母或符号。可以用简单的概率空间描述离散无记忆信源。设信源字母表为 $M=\{a_i, i=0,1,2,\cdots,N-1\}$，字母 a_i 出现的概率为 $p(a_i)\geqslant 0$，且 $\sum_{i=0}^{N-1} p(a_i)=1$。信源产生的任一长为 L 个符号的消息序列为 $m=(m_1,m_2,\cdots,m_L)$，$m_l\in M$，$1\leqslant l\leqslant L$。若我们研究的是所有长为 L 的信源输出，则称 $P=M^L=\{m=(m_1,m_2,\cdots,m_L)\mid m_l\in M, 1\leqslant l\leqslant L\}$ 为消息空间或明文空间，它含有 N^L 个元素。若信源为有记忆时，我们需要考虑 P 中各元素的概率分布。当信源为无记忆时，有 $p_P(m)=p(m_1,m_2,\cdots,m_L)=\prod_{l=1}^{L} p(m_l)$。信源的统计特性对密码的设计和分析有着重要影响。

密钥源是产生密钥序列的源。密钥通常是离散的，设密钥字母表为 $B=\{b_t, t=0,1,2,\cdots,s-1\}$。字母 b_t 的概率为 $p(b_t)\geqslant 0$，且 $\sum_{t=0}^{s-1} p(b_t)=1$。一般设计中使密钥源为无记忆均匀分布源，所以各密钥符号为独立等概。把长为 r 的密钥序列 $k=k_1k_2\cdots k_r, k_1, k_2, \cdots, k_r \in B$ 的全体称作密钥空间 K，且 $K=B^r$。一般地，消息空间与密钥空间彼此独立。合法的接收者知道 k 和密钥空间 K。窃听者不知道 k，也不知道或不确知 K。

加密变换是将明文空间中的元素 m 在密钥控制下变为密文 c，即 $c=(c_1,c_2,\cdots,c_{L'})=E_k(m_1,m_2,\cdots,m_L)$，称 c 的全体为密文空间，以 $C=Y^{L'}$ 表示之，其中 Y 表示密文字母表。通常密文字母集和明文字母集相同，因而一般有 $L=L'$。

密文空间的统计特性由明文和密钥的统计特性决定。下面我们来说明这一点。假定明文 $m\in P$ 发生的概率为 $p_P(m)$，密钥 k 被选择的概率为 $p_K(k)$，因为密钥是在发送者发送消息之前选定的，因而可假定消息空间和密钥空间是独立的。对一个密钥 $k\in K$，令 $C_k=\{E_k(m)\mid m\in P\}$，对每一个 $c\in C$，我们有

$$p_C(c)=\sum_{\{k\mid c\in C_k\}} p_K(k)p_P(D_k(c)) \tag{2.1.1}$$

又 $p_C(c\mid m)=\sum_{\{k\mid m=D_k(c)\}} p_K(k)$，所以由 *Bayes* 公式可得

$$p_P(m\mid c)=\frac{p_P(m)\sum_{\{k\mid m=D_k(c)\}} p_K(k)}{\sum_{\{k\mid c\in C_k\}} p_K(k)p_P(D_k(c))} \tag{2.1.2}$$

由式(2.1.1)和(2.1.2)知，知道明文空间和密钥空间的概率分布的任何人都能确定出密文空间的概率分布和明文空间关于密文空间的条件概率分布。可见，密文空间的统计特性由明文空间和密钥空间的统计特性完全决定。

在保密系统研究中，我们假定信道是无干扰的，因而对于合法接收者，由于他知道解密变换和密钥而易于从密文得到原来的消息 m，即 $m=D_k(c)=D_k(E_k(m))$。

在无干扰的条件下，假定密码分析者可以得到密文 c。而且一般假定密码分析者知道明文的统计特性、加密体制、密钥空间及其统计特性(这正是 *Kerckhoff* 假设)，但不知道加密截获的密文 c 所用的特定密钥。在 *Kerckhoff* 假设下，密码的安全性完全寓于秘密密钥之中。

2.1.2 熵及其基本性质

熵(*entropy*)能被视作信息或不确定性的一个数学度量，它是在所有可能的消息集合上的一个概率分布函数。其精确定义如下。

定义 2.1.1 设 $X=\{x_i\mid i=1,2,\cdots,n\}$，$x_i$ 出现的概率为 $p(x_i)\geqslant 0$，且 $\sum_{i=1}^{n} p(x_i)=1$。集 X 的熵定义为

$$H(X)=-\sum_{i=1}^{n} p(x_i)\log_2 p(x_i) \tag{2.1.3}$$

H(X)表示集 X 中事件出现的平均不确定性，或为确定集 X 中出现一个事件平均所需的信息量(观测之前)，或集 X 中每出现一个事件平均给出的信息量(观测之后)。在式

(2.1.3)中定义 $0 \cdot \log_a 0 = 0$。采用以 2 为底的对数时，相应的信息单位称作比特(*bit*)。

对某一 i 有 $p(x_i)=1$，对其它的 $j\neq i$，有 $p(x_j)=0$。反过来，若对某一 i 有 $p(x_i)=1$，对其它的 $j\neq i$，有 $p(x_j)=0$，直接可算得 $H(X)=0$。

由 Jensen 不等式可得

$$H(X)=-\sum_{i=1}^{n}p(x_i)\log_2 p(x_i)=\sum_{i=1}^{n}p(x_i)\log_2\frac{1}{p(x_i)}\leqslant\log_2\sum_{i=1}^{n}p(x_i)\times\frac{1}{p(x_i)}=\log_2 n$$

而且，当且仅当对一切 $1\leqslant i\leqslant n$，有 $p(x_i)=\frac{1}{n}$ 时等号成立。

注意，我们在上面的证明过程中应用 Jensen 不等式时实际上假定了对所有的 $1\leqslant i\leqslant n$，有 $p(x_i)>0$。如果对某些 i，有 $p(x_i)=0$ 时，上述不等式仍然成立，只不过此时是严格的小于 $\log_2 n$。

定理 2.1.2 $H(X,Y)\leq H(X)+H(Y)$，当且仅当 X 和 Y 统计独立时等号成立。

证明：由概率知识可知

$$p(x_i)=\sum_{j=1}^{m}p(x_iy_j),p(y_j)=\sum_{i=1}^{n}p(x_iy_j),1\leqslant i\leqslant n,1\leqslant j\leqslant m$$

则

$$\begin{aligned}H(X)+H(Y)&=-\Big(\sum_{i=1}^{n}p(x_i)\log_2 p(x_i)+\sum_{j=1}^{m}p(y_j)\log_2 p(y_j)\Big)\\&=-\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)\log_2 p(x_i)p(y_j)\end{aligned}$$

再结合式(2.1.4)可得

$$H(X,Y)-H(X)-H(Y)=\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)\log_2\frac{p(x_i)p(y_j)}{p(x_iy_j)}\leqslant\log_2\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_i)p(y_j)$$

(由 Jensen 不等式)$=\log_2 1=0$，即 $H(X,Y)\leqslant H(X)+H(Y)$。

等号成立当且仅当对所有的 $1\leqslant i\leqslant n$ 及 $1\leqslant j\leqslant m$，有 $\frac{p(x_i)p(y_j)}{p(x_iy_j)}=c$($c$ 为一常数)。又

$$\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)=\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_i)p(y_j)=1$$

所以 $c=1$，故等号成立当且仅当对所有的 $1\leqslant i\leqslant n$ 及 $1\leqslant j\leqslant m$，有 $p(x_iy_j)=p(x_i)p(y_j)$，即 X 和 Y 统计独立。

定理 2.1.3 $H(X,Y)=H(Y)+H(X|Y)=H(X)+H(Y|X)$

证明：下面只证 $H(X,Y)=H(Y)+H(X|Y)$，类似地可证 $H(X,Y)=H(X)+H(Y|X)$。由式(2.1.4)及概率的乘法公式可得

$$\begin{aligned}H(X,Y)&=-\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)\log_2 p(x_iy_j)=-\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)\log_2 p(y_j)p(x_i|y_j)\\&=-\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)\log_2 p(y_j)-\sum_{i=1}^{n}\sum_{j=1}^{m}p(x_iy_j)\log_2 p(x_i|y_j)\\&=-\sum_{j=1}^{m}p(y_j)\log_2 p(y_j)-\sum_{i=1}^{n}\sum_{j=1}^{m}p(y_j)p(x_i|y_j)\log_2 p(x_i|y_j)\\&=H(Y)+H(X|Y)\end{aligned}$$

由定理 2.1.2 和定理 2.1.3 可推得。

推论 2.1.1 $H(X|Y)\leqslant H(X)$，当且仅当 X 和 Y 统计独立时等号成立。

2.1.3 完善保密性

衡量一个保密系统的安全性有两种基本的方法，一种是计算安全性(computational security)，又称实际保密性(practical secrecy)，另一种是无条件安全性(unconditional security)，又称完善保密性(perfect secrecy)。一个密码系统说是计算上安全的，如果利用最好的算法(已知的或未知的)破译该系统需要至少 N 次运算，这里 N 是某一个确定的、很大的数。问题是还没有一个已知的实际密码系统能被证明是计算上安全的。在实际中，人们说一个密码系统是"计算上安全的"，意指利用已有的最好的方法破译该系统所需要的努力超过了敌手的破译能力(诸如时间、空间和资金等资源)或破译该系统的难度等价于解数学上的某个已知难题。当然，这只是提供了系统是计算上安全的一些证据，并没有真正证明系统是计算上安全的。一个密码系统说是无条件安全的，如果具有无限计算资源(诸如时间、空间、设备和资金等)的密码分析者也无法破译该系统。

保密系统的安全性是针对某种攻击而言的，本节我们针对唯密文攻击来研究保密系统的完善保密性，亦即无条件安全性。当然它不一定能保证在已知明文或选择明文攻击下也是完善保密的。Massey 曾将 Shannon 的唯密文攻击下的完善保密性概念推广到已知明文攻击下的情况，详见文献[7]。

研究密码系统的无条件安全性显然不能用计算复杂性的观点来研究，因为允许计算时间是无限的，一般采用概率的观点来研究密码系统的无条件安全性。

利用 2.1.2 节中介绍的熵的概念，可计算 2.1.1 节介绍的保密系统的各部分的熵。令明文空间的熵为 $H(P)=H(M^L)$，密钥空间的熵为 $H(K)=H(B^t)$，密文空间的熵为 $H(C)=H(Y^L)$，在已知密文条件下明文的含糊度为 $H(M^L|Y^L)$，在已知密文条件下密钥的含糊度为 $H(K|Y^L)$。从唯密文攻击来看，密码分析者的任务是从截获的密文中提取有关明文的信息 $H(M^L)-H(M^L|Y^L)$ 或从密文中提取有关密钥的信息 $H(K)-H(K|Y^L)$。可见，$H(M^L|Y^L)$ 和 $H(K|Y^L)$ 越大，窃听者从密文中能够提取出的有关明文和密钥的信息量就越小。因在已知密钥和密文的条件下明文信息已完全确定，所以 $H(M^L|Y^LK)=0$。再利用 2.1.2 节中的熵的基本性质可推知

$$\begin{aligned} H(M^L|Y^L) &\leqslant H(M^L|Y^L)+H(K|M^L,Y^L)=H(M^LK|Y^L) \\ &=H(K|Y^L)+H(M^L|Y^LK)=H(K|Y^L)\leqslant H(K) \end{aligned}$$

这样

$$H(M^L)-H(M^L|Y^L)\geqslant H(M^L)-H(K) \tag{2.1.8}$$

这表明，保密系统的密钥量越小，其密文中含有的关于明文的信息量就越大。至于密码分析者如何有效地提取这些信息是另外的问题。从密码系统的设计者的角度来看，自然要选择足够大的密钥量，而且希望从密文中提取的有关明文的信息尽可能地小，即 $H(M^L)-H(M^L|Y^L)$ 尽可能地小，一种特殊的情况是 $H(M^L)-H(M^L|Y^L)=0$。这正是下面将要定义的完善保密系统或无条件保密系统。

定义 2.1.3 一个保密系统 (P,C,K,ε,D) 称为是完善的或无条件的保密系统，如果 $H(P)=H(P|C)$。

由式(2.1.8)知，完善保密系统存在的必要条件是 $H(P)\leq H(K)$。可见，要构造一个完善保密系统，其密钥量的对数(密钥空间为均匀分布的条件下)必须不小于明文集的熵。

定理 2.1.4 完善保密系统存在。

证明：今以构造法证之。不失一般性可假定明文、密钥和密文都是二元数字序列，此时

$$P=\{m=(m_1,m_2,\cdots,m_L)\mid m_l\in Z_2,1\leqslant l\leqslant L\}$$
$$K=\{k=(k_1,k_2,\cdots,k_r)\mid k_i\in Z_2,1\leqslant i\leqslant r\}$$
$$C=\{c=(c_1,c_2,\cdots,C_{L'})\mid c_{l'}\in Z_2,1\leqslant l'\leqslant L'\}$$

今选定 $L=L'=r$，并假定 P 和 K 相互统计独立，k 是一个随机数字序列，即对一切 $k\in K$，有 $P_K(k)=1/2^L$。

加密变换为 $c=E_k(m)=m\oplus k$，式中的加法是逐位模 2 加，即 $c_l=m_l\oplus k_l$，$1\leqslant l\leqslant L$。

解密变换为 $m=D_k(c)=c-k$，式中的减法是逐位模 2 减，对于二元域，模 2 加与模 2 减一样。

要证明该保密系统是完善的，由推论 2.1.1 可知，只需证明对一切 $c\in C$ 及 $m\in P$，有 $p_P(m\mid c)=p_P(m)$（即明文空间与密文空间相互统计独立）即可。

设 $c\in C$，那么

$$p_C(c)=\sum_{k\in K}p_K(k)p_P(D_k(c))=\frac{1}{2^L}\sum_{k\in K}p_P(D_k(c))$$
$$=\frac{1}{2^L}\sum_{k\in K}p_P(c-k)$$

对固定的 c，当 k 取遍 K 时，$c-k$ 也取遍 P，所以对任何 $c\in C$，有 $p_C(c)=\frac{1}{2^L}$。又因为对每一个 $m\in P$ 及 $c\in C$，有唯一的一个密钥 $k\in K$ 使得 $E_k(m)=c$，所以

$$p_C(c\mid m)=p_K(c-m)=1/2^L$$

故由 Bayes 公式知

$$p_P(m\mid c)=\frac{p_P(m)p_C(c\mid m)}{p_C(c)}=p_P(m)$$

定理 2.1.4 中构造的系统在唯密文攻击下是安全的，但易受已知明文攻击，这是因为密钥 k 可由明文 m 和密文 c 进行模 2 加获得。这就要求每发送一条消息都要产生一个新的密钥并在一个安全的信道上传送，这同时也给密钥管理带来了严重的问题。因此，无条件安全保密系统是很不实用的，也具有很大的局限性，但在军事上和外交上很早就使用了这种体制，并且现在似乎在有些场合仍在使用。人们传统上称这种系统为“一次一密”。在实际中，人们设计一个密码系统的目标是希望一个密钥能用来加密一条相对长的明文串（也就是一个密钥能用来加密许多消息）并且至少是计算上安全的，诸如第 5 章中将要介绍的 DES 体制。

2.1.4 伪密钥和唯一解距离

本小节主要讨论在唯密文攻击下破译一个密码系统时密码分析者必须处理的密文量的理论下界。Shannon 从密钥含糊度 $H(K\mid C)$ 出发研究了这个问题，$H(K\mid C)$ 度量了给定密文下密钥的不确定性。首先，我们看一个定理，该定理反映了一个保密系统的各部分的熵之间的关系。

定理 2.1.5 设 (P,C,K,ϵ,D) 是一个保密系统，那么

$$H(K\mid C)=H(K)+H(P)-H(C)$$

证明：由定理 2.1.3 知

$$H(K,P,C) = H(C|K,P) + H(K,P) = H(P|K,C) + H(K,C)$$

由于密钥和明文唯一确定密文。密钥和密文唯一确定明文，密钥和明文统计独立，所以

$$H(C|K,P) = H(P|K,C) = 0, H(K,P) = H(K) + H(P)$$

从而

$$H(K) + H(P) = H(K,C)$$

又

$$H(K,C) = H(C) + H(K|C)$$

故

$$H(K|C) = H(K) + H(P) - H(C)$$

假定所使用的密码系统是(P,C,K,ε,D)。设$m\in P, k\in K$，则$c=E_k(m)\in C$。密码分析的基本目的是确定密钥。我们考虑唯密文攻击，并且假定分析者具有无限的计算资源。也假定分析者知道明文是某一自然语言，诸如英语。当分析者截获到c时，他用所有的密钥解密c得$m'=D_k(c), k\in K$，他记录所有有意义的消息m'对应的密钥，这些密钥作成的集合往往不只含一个元素(此集合至少含一个元素，因为正确的密钥在里边)，把那些是可能的但不正确的密钥称为伪密钥(spurious key)。

我们的目的是推导伪密钥的期望数的下界。为此，我们先定义两个概念，即语言的速率(rate)(或熵)和语言的多余度(redundancy)。

定义 2.1.4 假定L是一种自然语言，语言L的速率或熵定义为

$$H_L = \lim_{n\to\infty} \frac{H(A^n)}{n} \tag{2.1.9}$$

语言L的多余度定义为

$$R_L = 1 - \frac{H_L}{\log_2 |A|} \tag{2.1.10}$$

其中A表示L的字母集，$|A|$表示A中字母的个数，A^n表示所有明文n-字母报构成的全体。

H_L表示语言L的每个字母的平均信息比特数，R_L度量了“多余字母”的比例。

定理 2.1.6 假定(P,C,K,ε,D)是一个私钥保密系统，$|C|=|P|$。设R_L表示明文自然语言的多余度，那么给定一个充分长(长为n)的密文串，伪密钥的期望数$\overline{S_n}$满足

$$\overline{S_n} \geq \frac{2^{H(K)}}{|P|^{nR_L}} - 1 \tag{2.1.11}$$

证明：设明文字母集为A，明文空间为$P=A^L$，这里L为一正整数。给定$Y\in C^n$，定义$K(Y)=\{k\in K|$存在$x\in P^n$，使得$p_{P^n}(x)>0, E_k(x)=Y\}$，即$K(Y)$是在给定密文串Y的条件下可能的密钥的集合。如果Y是观察到的密文串，那么伪密钥的数是$|K(Y)|-1$，这是因为可能的密钥中只有一个是正确的。因此，伪密钥的期望数$\overline{S_n}$为

$$\overline{S_n} = \sum_{Y\in C^n} p(Y)(|K(Y)| - 1) = \sum_{Y\in C^n} p(Y)|K(Y)| - 1 \tag{2.1.12}$$

由定理 2.1.5 可知

$$H(K|C^n) = H(K) - H(P^n) - H(C^n)$$

易知

$$H(C^n) \leqslant n\log_2|C|$$

又当 n 充分大时，可由定义 2.1.4 对 $H(P^n)$进行估计，

$$H(P^n) = H(A^{nL}) \approx nLH_L = nL(1 - R_L)\log_2|A| = n(1 - R_L)\log_2|P|$$

所以

$$H(K|C^n) \geqslant H(K) - nR_L\log_2|P| \tag{2.1.13}$$

注意，已假定$|P| = |C|$。

而由 Jensen 不等式及式(2.1.12)可知

$$\begin{aligned} H(K|C^n) &= \sum_{Y\in C^n} p(Y)H(K|Y) \leqslant \sum_{Y\in C^n} p(Y)\log_2|K(Y)| \\ &\leqslant \log_2 \sum_{Y\in C^n} p(Y)|K(Y)| = \log_2(\overline{S_n} + 1) \end{aligned} \tag{2.1.14}$$

所以，由式(2.1.13)和式(2.1.4)可得式(2.1.11)。

定义 2.1.5 一个保密系统的唯一解距离定义为使得伪密钥的期望数等于零的 n 的值，记为 n_0，即在给定的足够的计算时间下分析者能唯一地计算出密钥所需要的密文的平均量。

在式(2.1.11)中，令$\overline{S_n}=0$，可得唯一解距离的一个近似估计，即 $n_0 \approx \dfrac{H(K)}{R_L\log_2|P|}$。这表明，当截获的密文量大于 n_0 时，原则上可以破译该密码。当截获的密文量小于 n_0 时，就存在有多种可能的密钥解，密码分析者无法从中确定哪一个解是正确的。当然，n_0 仅仅是一个理论值，一般破译密码系统所需要的密文量都远大于 n_0。

本节最后我们简要介绍一下 Shannon 在他的论文[1]中提到的设计密码的一些基本观点。

首先我们介绍 Shannon 的组合密码系统的观点。为了便于实现，实际应用中常常将较简单且易于实现的密码系统进行组合，构成较复杂的、密钥量较大的密码系统。Shannon 提出了两种可能的组合方法。一种是概率加权和方法，即以一定的概率随机地从几个子系统中选择一个用于加密当前的明文。设有 r 个子系统，以 $T_1, T_2, \cdots, T_r$ 表示，相应的被选用的概率为 $p_1, p_2, \cdots, p_r$，其中 $\sum_{i=1}^{r} p_i = 1$。其概率加密系统可表示成 $T = p_1T_1 + p_2T_2 + \cdots + p_rT_r$。显然，系统 T 的密钥量是各子系统的密钥量之和。另一种是乘积方法，诸如，设有两个子密码系统 T_1 和 T_2，则先以 T_1 对明文进行加密，然后再以 T_2 对所得结果进行加密。其中 T_1 的密文空间需作为 T_2 的“明文”空间。乘积密码可表示成 $T = T_1T_2$。类似地可定义任何有限个子密码系统的乘积。

其次我们介绍 Shannon 的挫败统计分析的观点。用统计分析法可以破译多种密码。为了挫败统计分析，Shannon 建议采用两种方法，一种是在加密之前将语言的一些多余度除去。诸如，在计算机系统中，在加密之前，采用 Huffman 编码[3]除去多余度来压缩一个文件。另一种是采用所谓的“扩散 (diffusion)”和“混淆 (confusion)”这两种加密技术扩散或混淆多余度。所谓扩散，就是将每一位明文数字的影响尽可能迅速地散布到较多个输出的密文数字中，以便隐蔽明文数字的统计特性。所谓混淆，就是将密文和明文的统计特性之间的关系复杂化。

2.2 Simmons 的认证系统的信息理论

认证码最早出现在文献[8]中，但 G. J. Simmons[9]在 1984 年首次系统地提出了认证系统的信息理论。他将信息论用于研究认证系统的理论安全性和实际安全性问题，指出认证系统的性能极限以及设计认证码所必须遵循的原则。虽然这一理论还不太成熟和完善，但它在认证系统中的地位与 Shannon 的信息理论在保密系统中的地位一样重要，它为认证系统的研究奠定了理论基础。

认证理论的主要研究目标有两个：一个是推导欺骗者欺骗成功的概率的下界；另一个是构造欺骗者欺骗成功的概率尽可能小的认证码，当然，也要考虑到其他因素，诸如信源集、消息集和编码规则集的大小等。目前研究的认证系统模型主要有两种：一种是无仲裁者的认证系统模型，在这种模型中，只有三种参加者，即消息的发送者、接收者和敌手，消息的发送者和接收者之间相互信任，他们拥有同样的秘密信息，共同来对付敌手，诸如文献[9～29]中研究的认证系统均为这种模型；另一种是有仲裁者的认证系统模型，在这种模型中，有四种参加者，即消息的发送者、接收者、敌手和仲裁者，消息的发送者和接收者之间相互不信任，但他们都信任仲裁者，仲裁者拥有所有的秘密信息并不进行欺骗，诸如文献[30～35]中研究的认证系统就是这种模型。

构成一个认证码的基本要素有三个，即信源集合、消息集合和编码规则集合，其中每一个编码规则由一个秘密密钥来控制。发送者的任何一个编码规则都是从信源集合到消息集合的一个映射，这个映射的每一个原像可能只有一个像也可能有几个像。如果发送者的每个编码规则的每一个原像都只有一个像，我们称这种认证码为无分裂的认证码。如果发送者的某个编码规则的某个原像有不只一个像，我们称这种认证码为有分裂的认证码。目前大部分文献中研究的均为无分裂的认证码，关于有分裂的认证码的研究可参阅文献[15,23,24]。信息的认证和保密是信息安全的两个不同方面，一个认证码可以具有保密功能，也可以没有保密功能，没有保密功能的认证码亦称 Cartesian 码。对具有保密功能的认证码而言，任何给定的消息没有给出关于信源的任何信息。而对 Cartesian 码而言，每个消息都独立于所使用的编码规则并唯一确定一个信源，无论谁看到消息都能断定这一消息所代表的信源。

在一个认证系统中，假设发送者与接收者在约定使用的编码规则后，连续发出 r 个消息，敌手在观察到这 r 个消息后，他可以分析得到关于当时所用的编码规则的一些信息，这时他发出消息 m，希望接收者能接收，我们称这种攻击为 r 阶欺骗攻击，这种认证系统通常称为 r 重认证系统。在这种意义下，文献[9]中研究的认证系统就是 1 重认证系统，o 阶欺骗攻击就是模仿攻击，1 阶欺骗攻击就是代替攻击。人们已对 r 重认证系统进行了大量研究，感兴趣的读者可参阅文献[7,12,16,22,23,24]。

研究认证理论的观点主要有两种：一种是利用组合论的观点；另一种是利用信息论的观点。限于篇幅，我们不能对现有的认证理论作全面的、深入的、系统的介绍。本节主要针对无分裂的、没有保密功能的、无仲裁者的认证系统模型，介绍 Simmons[9,26]发展的认证系统的信息理论。

2.2.1 认证系统的数学模型

一个无仲裁者的认证系统(authentication system without arbiter)模型如图 2.2.1 所示。在这个系统中,发送者和接收者之间相互信任,共同对付敌手。最终的目的是能使发送者通过一个公开的无干扰的信道将消息发送给接收者,接收者不仅能收到消息本身,而且还能验证消息是否来自发送者及消息是否被敌手窜改过。敌手不仅可截收和分析信道中传送的消息,而且可伪造消息发送给接收者进行欺诈,他不再像保密系统中的分析者那样始终处于消极被动地位,而是可发起主动攻击。

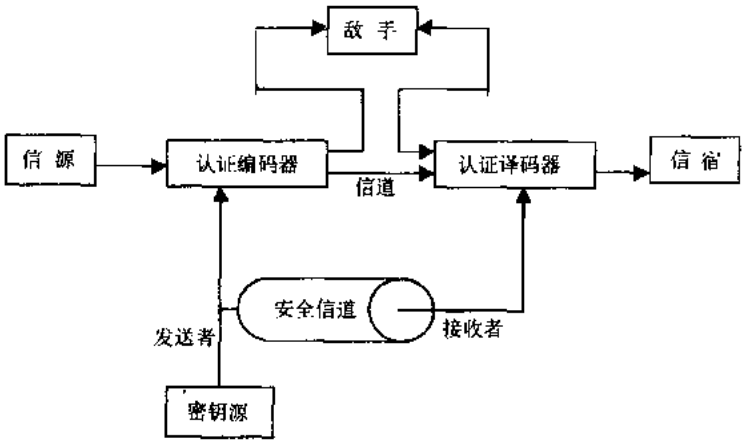


图 2.2.1 无仲裁者的认证系统模型

一个无分裂的、没有保密功能的、无仲裁者的认证系统可由满足下列条件的四重组(S,A,K,ε)来描述:

(1)S 是所有可能的信源状态(source state)作成的一个有限集,称为信源集;

(2)A 是所有可能的认证标签(authenticaiton tag)作成的一个有限集;

(3)K 是所有可能的密钥作成的一个有限集,称为密钥空间;

(4)对每一个 $k\in K$,有一个认证规则(authentication rule)$e_k\in\varepsilon:S\to A$。

消息(Message)集 M 定义为

$$M = S \times A$$

为了传送一个消息,发送者和接收者采纳下列协议。首先,他们共同秘密地选择一个随机密钥 $k\in K$;其次,假定发送者想在一个不安全的信道上给接收者传送一个信源状态 $s\in S$,发送者计算 $a=e_k(s)$并把消息(s,a)发送给接收者。当接收者收到(s,a)时,他计算 $a'=e_k(s)$,如果 $a'=a$,那么他把这个消息作为真消息接收,否则他拒绝接收。

在这里,我们只考虑两种不同类型的攻击,即模仿攻击(impersonation attack)和代替攻击(substituation attack)。在这些攻击中,敌手是一个中间入侵者。在模仿攻击中,敌手在信道中插入一个消息(s,a),希望接收者能把它作为真消息接收。在代替攻击中,敌手在信道中观察到一个消息(s,a)后,他将(s,a)改变为(s',a'),$s'\neq s$,希望接收者把它作为真消息接收。下面我们来推导这两种攻击成功的概率的计算公式。

设 p_{a_0}表示敌手采用模仿攻击时最大可能的欺骗接收者成功的概率,p_{a_1}表示在代替攻击时最大可能的欺骗接收者成功的概率。Simmons 将敌手欺骗接收者成功的概率定义

为

$$p_d = \max\{p_{d_0}, p_{d_1}\} \tag{2.2.1}$$

我们假定敌手知道所使用的认证码以及 S 和 K 的概率分布 p_S 和 p_K，只有发送者和接收者拥有的密钥的值 k 敌手不知道。

在认证理论的研究中，认证矩阵是一个很有用的工具，它是认证码的一种表示方式，这种表示对研究和理解问题会带来许多好处。一个认证矩阵是一个 $|K|\times|S|$ 阶矩阵，它的行由密钥来标记，它的列由信源状态来标记，对每一个 $k\in K$ 和 $s\in S$，该矩阵的第 k 行第 s 列的元素是 $e_k(s)$。

首先考虑 p_{d_0} 的计算。设 k_0 表示由发送者和接收者选择的密钥。对 $s\in S$ 和 $a\in A$，用 Payoff(s,a) 表示接收者将把消息 (s,a) 作为真消息接收的概率。易知

$$\text{Payoff}(s,a) = p(a = e_{k_0}(s)) = \sum_{\{k\in K | e_k(s)=a\}} p_K(k) \tag{2.2.2}$$

可见，Payoff(s,a) 能按如下方式进行计算：将认证矩阵的第 s 列中取值为 a 的这些行所对应的密钥的概率相加。

敌手为了使他成功的机会增大，他将选择 (s,a) 使得 Payoff(s,a) 最大，因此 p_{d_0} 的计算公式为

$$p_{d_0} = \max\{\text{Payoff}(s,a) | s\in S, a\in A\} \tag{2.2.3}$$

由式(2.2.2)可知，Payoff(s,a) 不依赖于概率分布 p_S，而只依赖于概率分布 p_K。所以 p_{d_0} 不依赖于概率分布 p_S，而只依赖于概率分布 p_K。

其次考虑 p_{d_1} 的计算。p_{d_1} 一般既依赖于概率分布 p_S 又依赖于概率分布 p_K。假定敌手在信道中观察到一个消息 (s,a)，他将用某一消息 (s',a') 来代替 (s,a)，$s'\neq s$。对 $s,s'\in S$，$s'\neq s$ 和 $a,a'\in A$，用 Payoff$(s',a';s,a)$ 表示用 (s',a') 来代替 (s,a) 欺骗接收者成功的概率，那么

$$\begin{aligned}\text{Payoff}(s',a';s,a) &= p(a' = e_{k_0}(s') | a = e_{k_0}(s)) = \frac{p(a' = e_{k_0}(s'), a = e_{k_0}(s))}{p(a = e_{k_0}(s))} \\ &= \frac{\sum_{\{k\in K | e_k(s)=a, e_k(s')=a'\}} p_K(k)}{\sum_{\{k\in K | e_k(s)=a\}} p_K(k)} = \frac{\sum_{\{k\in k | e_k(s)=a, e_k(s')=a'\}} p_K(k)}{\text{Payoff}(s,a)}\end{aligned} \tag{2.2.4}$$

式(2.2.4)中的分子可按下述方式进行计算：将认证矩阵的第 s 列中取值为 a 同时在第 s' 列中取值为 a' 的这些行所对应的密钥的概率相加。

因为敌手想极大化他欺骗接收者成功的机会，所以他将计算

$$p_{s,a} = \max\{\text{Payoff}(s',a';s,a) | s'\in S, s'\neq s, a'\in A\} \tag{2.2.5}$$

$p_{s,a}$ 表示在敌手观察到消息 (s,a) 后，他用代替攻击欺骗接收者成功的概率。

显然，我们可将 $p_{s,a}$ 在消息集 $M = S\times A$ 上的期望值作为欺骗概率 p_{d_1} 的值。设 p_M 是消息集 M 上的概率分布，则

$$p_{d_1} = \sum_{(s,a)\in M} p_M(s,a) p_{s,a} \tag{2.2.6}$$

概率分布 p_M 可由下式计算出：

$$p_M(s,a) = p_S(s) \times p_K(a|s) = p_S(s) \times \sum_{(k\in K|e_k(s)=a)} p_K(k)$$

$$= p_S(s) \times \text{Payoff}(s,a) \tag{2.2.7}$$

这样

$$p_{d_1} = \sum_{(s,a)\in M} p_S(s) \times \text{Payoff}(s,a) \times \max\{\text{Payoff}(s',a';s,a) | s' \in S, s' \neq s, a' \in A\}$$

$$= \sum_{(s,a)\in M} p_S(s) \max\left\{ \sum_{(k\in K|e_k(s)=a, e_k(s')=a')} p_K(k) | s' \in S, s' \neq s, a' \in A \right\}$$

设

$$q_{s,a} = \max\left\{ \sum_{(k\in K|e_k(s)=a, e_k(s')=a')} p_K(k) | s' \in S, s' \neq s, a' \in A \right\}$$

则

$$p_{d_1} = \sum_{(s,a)\in M} p_S(s) q_{s,a} \tag{2.2.8}$$

最后用一个例子来说明 p_{d_0}和 p_{d_1}的计算。

例 2.2.1 设 $S=\{1,2,3,4\}$, $K=\{1,2,3\}$, $A=\{1,2\}$, $p_S(i)=\frac{1}{4}$, $1\leqslant i\leqslant 4$, $p_K(1)=\frac{1}{2}$, $p_K(2)=p_K(3)=\frac{1}{4}$, $\varepsilon=\{e_1,e_2,e_3\}$, 其中三个认证规则分别为

$$e_1: S \to A, e_1(1) = e_1(2) = e_1(3) = 1, e_1(4) = 2$$

$$e_2: S \to A, e_2(1) = e_2(2) = e_2(4) = 2, e_2(3) = 1$$

$$e_3: S \to A, e_3(1) = e_3(4) = 1, e_3(2) = e_3(3) = 2$$

该码的认证矩阵为

密钥 \ 信源状态	1	2	3	4
1	1	1	1	2
2	2	2	1	2
3	1	2	2	1

Payoff(s,a)的值如下：

$$\text{Payoff}(1,1)=\text{Payoff}(3,1)=\text{Payoff}(4,2)=\frac{3}{4}$$

$$\text{Payoff}(2,1)=\text{Payoff}(2,2)=\frac{1}{2}$$

$$\text{Payoff}(1,2)=\text{Payoff}(3,2)=\text{Payoff}(4,1)=\frac{1}{4}$$

因此，由式(2.2.3)可知，$p_{d_0}=\frac{3}{4}$。敌手的最优的模仿策略是在信道上放消息(1,1),(3,1)或(4,2)中的任何一个。

现在我们来计算 p_{d_1}。先计算 Payoff$(s',a';s,a)$，它的值见表 2.2.1. 这样，我们有 $p_{1,1}=\frac{2}{3}$, $p_{2,2}=\frac{1}{2}$, $p_{s,a}=1$, $(s,a)\neq(1,1),(2,2)$。由式(2.2.7)和(2.2.6)可计算出 $p_{d_1}=\frac{7}{8}$。敌手的一个最优的策略如下：

$(1,1)\to(2,1)$ $(1,2)\to(2,2)$ $(2,1)\to(1,1)$ $(2,2)\to(1,1)$

$(3,1)\to(4,2)$ $(3,2)\to(1,1)$ $(4,1)\to(1,1)$ $(4,2)\to(3,1)$

这个策略的确使得 $p_{d_1}=\frac{7}{8}$。当然也可以利用式(2.2.8)计算 p_{d_1}。

表 2.2.1

(s,a) \ (s',a')	(1,1) (1,2)	(2,1) (2,2)	(3,1) (3,2)	(4,1) (4,2)
(1,1)		2/3 1/3	2/3 1/3	1/3 2/3
(1,2)		0 1	1 0	1 0
(2,1)	1 0		0 1	0 1
(2,2)	1/2 1/2		1/2 1/2	1/2 1/2
(3,1)	2/3 1/3	2/3 1/3		0 1
(3,2)	1 0	0 1		1 0
(4,1)	1 0	0 1	0 1	
(4,2)	2/3 1/3	2/3 1/3	1 0	

2.2.2 认证码的信息论下界

我们已经看到认证码的安全性由欺骗概率 p_{d_0}和 p_{d_1}来度量。因此我们想构造使得这些概率尽可能小的认证码，但还需考虑其它因素。在构造一个认证码时，我们期望它达到以下几个目标：

(1)欺骗概率 p_{d_0}和 p_{d_1}必须足够小，以便获得期望的安全水平；

(2)信源状态的数目必须足够大，以便我们能通过在一个信源状态后附加一个标签来传送期望的消息；

(3)密钥空间的尺寸尽可能小，因为密钥的值必须在一个安全的信道上传送。

本小节我们主要利用信息论的观点，在认证码的其它参数确定的条件下推导欺骗概率的下界。

定理 2.2.1 设(S,A,K,ε)是一个认证码，则

$$p_{d_0}\geqslant 2^{H(K|M)-H(K)} \tag{2.2.9}$$

等号成立，当且仅当对任何存在 $e\in\varepsilon$，使得 $e(s)=a$ 的 $m=(s,a)\in M=S\times A$，Payoff(s,a) =常数，且与消息 $m=(s,a)\in M=S\times A$ 的取值无关。

证明：因为 Payoff(s,a)的最大值不小于它们的期望值，所以由式(2.2.3)可知

$$p_{d_0}\geqslant\sum_{s\in S,a\in A}p_M(s,a)\text{Payoff}(s,a) \tag{2.2.10}$$

由 Jensen 不等式可知

$$\begin{aligned}\log_2 p_{d_0}&\geqslant\log_2\sum_{s\in S,a\in A}p_M(s,a)\text{Payoff}(s,a)\\&\geqslant\sum_{s\in S,a\in A}p_M(s,a)\log_2\text{Payoff}(s,a)\end{aligned} \tag{2.2.11}$$

由式(2.2.7)和(2.2.11)可知

$$\log_2 p_{d_0}\geqslant\sum_{s\in S,a\in A}p_S(s)p_A(a|s)\log_2 p_A(a|s)=-H(A|S) \tag{2.2.12}$$

因为 K 和 S 是相互独立的并且密钥和信源状态唯一确定认证标签，所以

$$H(K,S) = H(K) + H(S), H(A|K,S) = 0$$

又

$$H(K,A,S) = H(K|A,S) + H(A|S) + H(S) = H(A|K,S) + H(K,S)$$

所以

$$-H(A|S) = H(K|A,S) - H(K) = H(K|M) - H(K)$$

故 $p_{d_0} \geqslant 2^{H(K|M)-H(K)}$。

式(2.2.11)中等号成立，当且仅当 Payoff(s,a)=常数，且与消息$(s,a) \in M$ 的取值无关。这也是式(2.2.10)中等号成立的充分条件。

定理 2.2.1 表明 p_{d_0}不可能等于零，而且要有效地阻止模仿攻击需消息提供(至少原则上)许多有关密钥的信息，即部分密钥必须专用于提供认证性而不是保密性。

由式(2.2.1)和(2.2.9)可得

$$p_d \geqslant 2^{H(K|M)-H(K)} \tag{2.2.13}$$

此时使式(2.2.9)中等号成立的充要条件只是(2.2.13)中等号成立的必要条件，而非充分条件。

由于认证系统不能实现完全保护，所以将完善认证(perfect authentication)定义为给定认证码空间能使欺骗概率 p_d最小的认证系统。即使 $p_d=1$，也有完善认证可言。

定义 2.2.1 完善认证是使式(2.2.13)中等号成立的认证系统。

显然，每一个具有 $H(K|M)=H(K)$的认证系统提供平凡的完善认证性。

定理 2.2.2 完善认证系统存在。

证明：今以构造法证之。设 $S=\{0,1\}$，取一个正偶数 N，

$$A = Z_2^{N/2} = \{(a_1,a_2,\cdots,a_{N/2}) | a_i \in Z_2, 1 \leqslant i \leqslant N/2\}$$

$$K = Z_2^N = \{(k_1,k_2,\cdots,k_N) | k_i \in Z_2, 1 \leqslant i \leqslant N\}$$

由 $k=(k_1,k_2,\cdots,k_{N/2},k_{N/2+1},\cdots,k_N)$决定的编码规则 e_k 为：当 $s=0$ 时，$e_k(s)=(s,k_1,k_2,\cdots,k_{N/2})$；当 $s=1$ 时，$e_k(s)=(s,k_{N/2+1},\cdots,k_N)$。此时，$M=S\times A$，并假定所有 2^N 个密钥都是等概的。下面我们说明该认证码(S,A,K,ε)是完善的。

由于密钥被等概地选取，所以 $p_{d_0}=2^{-N/2}$，$p_{d_1}=2^{-N/2}$。从而 $p_d=2^{-N/2}$。又

$$H(K|M) - H(K) = N/2 - N = -N/2$$

所以

$$p_d = 2^{-N/2} = 2^{H(K|M)-H(K)}$$

即系统提供完善认证性。

现在我们用定理 2.2.1 来推导无分裂的 Cartesian 认证码的(S,A,K,ε) 的 r 阶欺骗概率 p_r 的信息论下界，即敌手连续获得 r 个消息后攻击成功的概率 p_r 的信息论下界。

定理 2.2.3 设(S,A,K,ε)是一个认证码，则

$$p_r \geqslant 2^{H(K|M^{r-1})-H(K|M^r)}, r = 0,1,2,\cdots \tag{2.2.14}$$

上述等号成立，当且仅当敌手获取 r 个消息 $m^r=(m_1,m_2,\cdots,m_r)(m_i=(s_i,a_i),1\leqslant i\leqslant r)$后，对任何存在 $e\in\varepsilon$，使得 $e(s)=a$ 的 $m=(s,a)\neq m_i(1\leqslant i\leqslant r)$，用 m 攻击成功的概率

$$p(m|m^r) = \sum_{e\in\varepsilon} X(e,m^r,m)p(e|m^r) = 常数$$

且与 m,m^r 的取值无关。其中 $M=S\times A$。$X(e,m^r,m)$定义为：如果 $e(s_i)=a_i(1\leqslant i\leqslant r)$且 $e(s)=a$，则 $X(e,m^r,m)=1$；否则，$X(e,m^r,m)=0$，p_r 定义为

$$p_r=\sum_{m^r\in M^r}p(m^r)\max_{m\in M}p_r(m|m^r)$$

证明：若敌手在信道中获得 $m^r=(m_1,m_2,\cdots,m_r)$，则他得到编码规则集 ε 的一个新的后验概率分布 $p(e_i|m^r),i=1,2,\cdots,n=|\varepsilon|$，而且得到消息集 $M=S\times A$ 的一个新的后验概率分布：$p(m|m^r),m\in M,m\neq m_i(1\leqslant i\leqslant r)$。若 $m=m_i,i=1,2,\cdots,r$，则 $p(m|m^r)=0$。由定理 2.2.1 可知，敌手获得 m^r 后攻击成功的概率 $p_r(m^r)$为

$$\log_2 p_r(m^r)\geqslant\sum_{m\in M}H(K|m^r,m)p(m|m^r)-H(K|m^r)$$

$$p(m^r)\log_2 p_r(m^r)\geqslant\sum_{m\in M}H(K|m^r,m)p(m,m^r)-H(K|m^r)p(m^r)$$

$$\sum_{m^r\in M^r}p(m^r)\log_2 p_r(m^r)\geqslant\sum_{m^r\in M^r}\sum_{m\in M}H(K|m^r,m)p(m^r,m)-\sum_{m^r\in M^r}H(K|m^r)p(m^r)$$

由 Jensen 不等式可知

$$\log_2\sum_{m^r\in M^r}p(m^r)p_r(m^r)\geqslant\sum_{m^r\in M^r}p(m^r)\log_2 p_r(m^r)\geqslant H(K|M^{r+1})-H(K|M^r)$$

要使等式成立，上面推导的每一步都必须取等号，因此，很容易得到定理中等号成立的充要条件。

值得注意的是，可用证明定理 2.2.3 的方法直接证明文献[16,22,23,24]等中有关认证码的欺骗概率 p_r 的信息论下界，这样做要比这些文献中的证明简单得多。

认证系统的安全性与保密系统的安全性一样也分为两类，即无条件安全性和实际安全性(practical security)，只是该领域的研究者习惯上将无条件安全性称为理论安全性(theoretical security)。

理论安全性与敌手的计算能力或时间无关，也就是说敌手破译体制所做的任何努力都不会优于随机地选择来碰运气。

实际安全性是根据破译认证体制所需的计算量来评价其安全性。实际安全性又分为计算安全性和可证明安全性两种。如果破译一个系统在原理上是可能的，但利用所有已知的算法和现有的计算工具不可能完成所要求的计算量，就称其为计算上安全的。如果能够证明破译某体制的困难性等价于解决某个数学难题，就称其为可证明安全的。这两种安全性虽都是从计算量来考虑，但不尽相同，计算安全要算出或估计出破译它的计算量下限，而可证明安全则要从理论上证明破译它的计算量不低于解已知难题的计算量。

2.3 注记和文献

Shannon[1]最早将信息论用于研究密码问题，在他的经典著作中除了本章介绍的一些概念和观点外，还详细论述了相似系统、密文等价类、纯密码、随机密码、理想保密系统等概念和观点。这篇论文至今对密码学的研究仍具有重要的指导意义。

关于信息论方面的知识，本章只作了一点简单介绍，感兴趣的读者可参阅文献[3～6]。

Hellman[37]曾对 Shannon 的随机密码体制做了一点推广。Massey[7]对 Shannon 的这一理论做了清晰的阐述。文献[38,39,40,41]研究了消息含糊度的界。文献[42]用唯一解距离分析了古典密码的破译问题。文献[43]讨论了密码设计中如何降低消息的多余度问题。

Simmons[9]首次系统地提出了认证系统的信息理论，它是研究认证系统和构造认证码的重要理论基础。关于认证理论的研究在 2.2 节的开头一段做了简要介绍，有兴趣的读者可进一步参阅有关文献。

关于认证理论的综述性文章可参阅文献[10,26,36]，其中文献[10]对文献[9]做了很好的解释，同时对 Simmons 下界给出了一个简单的证明。

值得指出的是，保密性和认证性是密码系统的两个无关属性，切勿以为用来提供一种属性的系统会自然而然地提供另一种属性。无保密性的完善认证和具有完善保密性的完善认证都是可能的。

参 考 文 献

[1] Shannon, C. E., Communication Theory of Secrecy System, Bell Syst. Tech. J., Vol. 28, pp. 656—715, 1949.

[2] Shannon, C. E., A mathematical Theory of Communication, Bell Syst. Tech. J., Vol. 27, pp. 379—423 & 623—656, 1948.

[3] Gallager, R. G., Information Theory and Reliable Communication, New York, Wiley, 1968.

[4] 王育民、梁传甲，信息与编码理论，西北电讯工程学院出版社，西安，1986.

[5] 常迥，信息理论基础，清华大学出版社，北京，1993.

[6] 周炯磐，信息理论基础，人民邮电出版社，北京，1983.

[7] Massey, J. L., Cryptology- A Selective Survey (Tutorial), in Digital Communications, Ed. by Biglier, E. & Prati, G., pp. 3—24, Netherland, 1986.

[8] Gilbert, E. N., Marwilliams, F. J. and Sloane, N. J., Codes which Detect Deception, Bell System Technical Journal, 1974, 53, pp. 405—424.

[9] Simmons, G. J., Authentication Theory / Coding Theory, Advances in Cryptology-Crypto'84, Spring-Verlag, 1985, pp. 411—431.

[10] Massey, J. L., Contemporary Cryptology, an Introduction, in Contemporary Cryptology, the Science of Information Integrity, Simmons, G. J. Ed., New York: IEEE Press, 1992, pp. 3—39.

[11] Johannesson, R. and Sgarro, A., Strengthening Simmon's bound on Impersonation, IEEE Trans. Inform. Theory, Vol. 37, pp. 1182—1185, July, 1991.

[12] Smeets, B., Bounds on the Probablity of Deception in Multiple Anuthentication, IEEE Trans. Inform. Theory, Vol. 40, No. 5, pp. 1586—1591, 1994.

[13] Wan, Z. X., Construction of Cartesian Authentication Codes from Unitary Geometry, Designs, Codes and Cryptography, No. 2, 1992, pp. 333—356.

[14] De Soete, M., Some Constructions for Authentication/ Secrecy Codes, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1988, pp. 57—75.

[15] De Soete, M., New Bounds and Constructions for Authentication /Secrecy Codes with Splitting, J. Cryptology, Vol. 3, No. 3, pp. 173—186, 1991.

[16] Walker, M., Information-theoretic Bounds for Authentication Schemes, J. Cryptology, Vol. 2, No. 2, 1990, pp. 131—143.

[17] Smeets, B., Vanroose, P. and Wan. Z. X., On the Construction of Authentication Codes with Secrecy and Codes

withstanding Spoofing Attacks of Order L≥2, Advances in Cryptology-Eurocrypt'90, 1991, pp. 306—312.

[18] Johannsson, T., A Shift Register Construction of Unconditionally Secure Authentication Codes, Designs, Codes and Cryptography, No. 4, 1994, pp. 69—81.

[19] Stinson, D. R., Some constructions and Bounds for Authentication Codes, J. Cryptology, No. 1, 1988, pp. 37—51.

[20] Stinson, D. R., The Combinatorics of Authentication and Secrecy Codes, J. Cryptography, No. 2, 1990, pp. 23—49.

[21] Stinson, D. R., Combinatorial Characterizations of Authentication Codes, Designs, Codes and Cryptography, No. 2, 1992, pp. 175—187.

[22] Rosenboaum, U., A Lower Bound on Authentication After Having Observed a Sequence of Message, J. Cryptology, No. 2, 1993, pp. 135—156.

[23] Sgarro, A., Lower Bound for Authentication Codes with Splitting, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 283—293.

[24] Pei, D. Y., Information-Theoritic Bounds for Authentication Codes and PBIB, J. Cryptology, Vol. 8, 1995, pp. 177—188.

[25] Safavi-Naini, R. and Tombak, Optimal Authentication Systems, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, 1994, pp. 12—27.

[26] Simmons, G. J., A Survey of Information Authentication, in Contemporary Cryptology, The Science of Information Integrity, pp. 379—419, IEEE Press, 1992.

[27] 万哲先、冯荣权，利用伪辛几何构作 Cartesian 认证码，Advances in Cryptology-Chinacrypt'94，科学出版社，1994，82—86 页.

[28] 裴定一、王学理，基于特征 2 的有限域圆锥曲线上的加密认证码的编码规则，Advances in Cryptology-Chinacrypt'96，科学出版社，1996，116—121 页.

[29] 冯登国、裴定一，一种无条件安全认证码的构造方法，电子学报，Vol. 25, No. 4, 1997, 138—139 页.

[30] Simmons, G. J., A Cartesian Product Construction for Unconditionally Secure Authentication Codes That Permit Arbitration, J. Cryptology, Vol. 2, No. 2, pp. 77—104, 1990.

[31] Simmons, G. J., Message Authentication with Arbitration of Transmitter/Receiver Disputes, Advances in Cryptology-Eurocrypt'87, Springer-Verlag, 1988, pp. 151—165.

[32] Brickell, E. F. and Stinson, D. R., Authentication Codes with Multiple Arbiters, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1988, pp. 51—55.

[33] Desmedt, Y. and Yung, M., Arbitrated Unconditionally Secure Authentication Can Be Unconditionally Protected Against Arbiter's Attack, Advances in Cryptology- Crypto'90, Springer-Verlag, 1991, pp. 177—188.

[34] Johasson, T., Lower Bounds on the Probability of Deception in Authentication with Arbitration, IEEE Trans. Inform. Theory, Vol. 40, No. 5, pp. 1573—1585, 1994.

[35] Kurosawa, K., New Bound on Authentication Code with Arbitration, Advances in Cryptology-Crypto'94, Springer-Verlag, 1994, pp. 140—149.

[36] 裴定一，认证码及其构造的一些研究，Advances in Cryptology-Chinacrypt'92，科学出版社，1992，66—73 页.

[37] Hellman, M. E., An Extension of the Shannon Theory Approach to Cryptography, IEEE Trans. on IT-23, No. 3, pp. 289—294, 1977.

[38] Blom, R. J., Bounds on Key Equivocation for Simple Substitution Cipher, IEEE Trans. on IT-25, No. 1, pp. 8—18, 1979.

[39] Blom, R. J., An Upper Bound on the Key Equivocation for Pure Cipher, IEEE Trans. On IT-30, No. 1, pp. 82—84, 1984.

[40] Dunham, J. G., Bounds on Message Equivocation for Simple Substitution Ciphers, IEEE Trans. On IT-26, No. 5, pp. 522—527, 1980.

[41] Von Tilburg, J. and Boekee, D., Divergence Bounds on Key Equivocation and Error Probability in Cryptanalysis, Advances in Cryptology-Crypto'85, pp. 489—513, Springer-Verlag, 1986.

[42] Deavours, C. A. , Cryptology Yesterday, Today, and Tomorrow, Artech House, 1987.

[43] Lu, S. C. , Randon Ciphering Bound on a Class Secrecy Systems and Discrete Message Sources, IEEE Trans. On IT-25, No. 4, pp. 405—414, 1979.

[44] Stinson, D. R. , Cryptography Theory and Practice, CRC Press, 1995.

[45] 王育民、何大可，保密学——基础与应用，西安电子科技大学出版社，西安，1990.

第 3 章　密码学的复杂性理论基础

现有的密码学著作关于复杂性理论的讨论，主要包括两个方面，即算法复杂性(complexity of algorithm)与问题复杂性(complexity of problem)。算法与问题复杂性为分析不同密码技术和算法的"计算复杂性"提供了一种方法。它对密码算法和技术进行比较，并确定它们的安全性。它是密码分析技术中分析计算需求和研究破译密码固有困难性的基础。但在某些密码协议和算法中仅考虑它的计算复杂性还是不够的，例如在一个身份识别协议中，要求用户 A 向用户 B 证明他的身份，那么如何来确保用户 A 在向用户 B 证明其身份的过程中无任何有用的信息泄漏呢？零知识证明(zero-knowledge proof)理论为我们解决这一问题提供了理论基础。在密码学中，零知识证明理论主要用于各种密码协议、密钥分配方案以及其它公钥体制等的设计和分析之中，它为所设计的系统的安全性提供了有力的证据。鉴于此，我们将零知识证明理论也归纳在复杂性理论中。

3.1　算法与问题复杂性理论

3.1.1　算法与问题

本小节我们来非正式地描述一下算法与问题这两个术语。

问题(problem)是指需要回答的一般性提问，通常含有若干个未定参数或自由变量。问题的描述包括两个方面：(1)给定所有的未定参数的一般性描述；(2)陈述"答案"或"解"必须满足的性质。若给问题的所有未知参数指定了具体的值，就得到该问题的一个实例(instance)。

例 3.1.1　在二元域 Z_2 上求解下列布尔函数组

$$\begin{cases} f_1(x_1,x_2,\cdots,x_n)=0 \\ f_2(x_1,x_2,\cdots,x_n)=0 \\ \cdots\cdots \\ f_m(x_1,x_2,\cdots,x_n)=0 \end{cases}$$

此问题的参数的集合为 $\{f_i(x_1,x_2,\cdots,x_n)\}_{i=1}^m$。此问题的解是向量 $(u_1,u_2,\cdots,u_n)\in Z_2^n$，它使得对一切 $1\leqslant i\leqslant m$，有 $f_i(u_1,u_2,\cdots,u_n)=0$。取 $m=n=3$，$f_1=x_1+x_2x_3$，$f_2=1+x_1x_2$，$f_3=1+x_1+x_2+x_3+x_1x_2+x_1x_2x_3$，就得到该问题的一个实例。

算法(algorithm)是指求解某个问题的一系列具体步骤。通常可理解为求解某个问题的通用计算机程序。算法总是针对某个问题来说的，例如，欧几里德算法用来求整数 a 和 b 的最大公因子，高斯消去法用来求解线性方程组。但针对一个问题的算法可以有好多种，例如，求解任意域上的线性方程组的算法就有好几种，诸如高斯消去法、Coppersmith-Winograd 消去法等。

如果一个算法能解答一个问题的任何实例，那么就说这个算法能解答这个问题。如果

针对一个问题至少存在一个算法可解答这个问题，那么就说这个问题是可解的(resolvable)，否则就说这个问题是不可解的(unresolvable)。

3.1.2 算法复杂性

一个算法的复杂性由该算法所需求的最大时间(T)和存储空间(S)度量。由于算法用于同一问题的不同规模实例所需求的时间和空间往往不同，所以总是将它们表示成问题实例的规模 n 的函数，其中 n 表示描述该实例所需的输入数据长度。一个算法用于某个问题的具有相同规模 n 的不同实例，其时间和空间需求也可能会有很大差异，所以有时需要研究其平均时间复杂性函数 $\overline{T}(n)$ 和平均空间复杂性函数 $\overline{S}(n)$，它们分别表示这种算法解该问题的规模为 n 的所有实例的时间和空间需求的平均值。

一个算法的复杂性通常用称之为"大 O"的符号来表示，它表示了算法复杂性的数量级。$f(n)=O(g(n))$ 意味着存在一个常数 c 和 n_0 使得对一切 $n\geqslant n_0$，有 $f(n)\leqslant c|g(n)|$。例如，假定 $f(n)=17n+10$，则 $f(n)=O(n)$，因为取 $g(n)=n$，$c=18$，$n_0=10$ 时，当 $n\geqslant n_0$ 时，有 $f(n)\leqslant cg(n)$。若 $f(n)$ 是 n 的一个 t 次多项式，即 $f(n)=a_tn^t+a_{t-1}n^{t-1}+\cdots+a_1n+a_0$，其中 t 为常数，则 $f(n)=O(n^t)$，即所有常数和低阶项都可忽略不计。

用数量级来度量一个算法的时间和空间复杂性，其优点在于它与所用的处理系统无关，因此它不需要知道不同指令的确切运行时间，或用于表示不同类型的数据所用的比特数，甚至连处理器的速度也不必知道。同时，它使我们能看出时间和空间的需求是怎样被输入数据的长度所影响。例如，如果 $T=O(n)$，那么输入长度加倍，算法的运行时间也加倍。如果 $T=O(2^n)$，那么输入长度增加 1 比特，算法的运行时间加倍。

通常按时间(或空间)复杂性对算法进行分类。多项式时间算法(Polynomial time algorithm)是指时间复杂性为 $O(n^t)$ 的算法，其中 t 为常数，n 是输入数据长度。若 $t=0$，就称它是常量(constant)的；若 $t=1$，就称它是线性(linear)的；若 $t=2$，就称它是二次(quadratic)的等等。指数时间算法(exponential time algorithm)是指时间复杂性为 $O(t^{h(n)})$ 的算法，其中 t 为常量，$h(n)$ 是一个多项式，当 $h(n)$ 大于常数而低于线性函数时，诸如时间复杂性为 $O(n^{\log_2 n})$，$O(e^{\sqrt{n\ln n}})$ 的算法，称为超多项式时间算法(superpolynomial time algorithm)。

表 3.1.1 不同类型算法运行时间

类 别	复 杂 性	$n=10^6$ 时的运算次数	实际时间
多项式			
常数	$O(1)$	1	1 微妙
线性	$O(n)$	10^6	1 秒
二次	$O(n^2)$	10^{12}	11.6 天
三次	$O(n^3)$	10^{18}	32,000 年
指数	$O(2^n)$	$10^{301,030}$	约 3×10^{301016} 年

当 n 很大时，不同类型的算法的复杂性可能会差别极大。例如，考察每微秒(us)能执行一条指令，即每秒 10^6 条指令，或每天 8.64×10^{10} 条指令的机器。表 3.1.1 给出了不同类型的算法在 $n=10^6$ 时的运行时间。当 $O(n^3)$ 时，在串行机上执行算法在计算上变得不可行了。然而，可以设想采用一百万个处理器的机器就可在大约 11.6 天内完成计算。对于

$T=O(2^n)$，即使我们采用上万亿个处理器并行工作，要执行这类算法在计算上也是不可行的。

许多密码可以采用穷举搜索整个密钥空间来破译，即尝试每个可能的密钥断定它是否可解密为有意义的明文或某个已知明文。若 $n=2^{H(K)}$ 是密钥空间的大小，则这种破译法的运行时间为 $T=O(n)=O(2^{H(K)})$。因此，按密钥量计算，时间是线性的，但以密钥长度算它是指数的。这就是为什么将 DES(见第 5 章)的密钥长度从 56 比特加倍为 112 比特时，虽然它的唯一解距离只增加一倍，但破译难度却大大增加。

3.1.3 问题复杂性

问题复杂性理论利用算法复杂性理论作为工具，将大量典型的问题按求解的代价进行分类。在问题复杂性理论中，与密码学关系最密切的是关于 NP 与 NP 完全问题的理论。

为了介绍问题的复杂性理论，我们先看一个概念——图林机(Turing machine)。图林机是一种假想计算机，它是一种具有无限读写能力的有限状态机，并且可以做无限的并行操作。图林机分为确定型和非确定型两种。所谓确定型图林机是指图林机的每一步的操作结果是唯一确定的。所谓非确定型图林机是指图林机的每一步的操作结果及下一步操作都有多种选择，不是唯一确定的。

一个问题的复杂性由在图林机上解此问题的最难的实例所需的最小时间与空间决定。一个问题的复杂性可理解为由解该问题的最有效的算法所需的时间与空间来度量。

在确定型图林机上可用多项式时间可解的问题，称为易处理的(tractable)。易处理的问题的全体称为确定性多项式时间可解类，记为 P。在确定型图林机上不能用多项式时间系统地解出的问题，称为难处理的(intractable)。在图林机上，没有办法写出求解算法的问题，称为不可判定的(undecidable)。在图林机上，可用“是”或“否”回答的问题或可求解的问题，称为可判定的(decidable)。

在非确定型图林机上可用多项式时间可解的问题，称为非确定性多项式时间可解问题，简称 NP 问题。NP 问题的全体称为非确定性多项式时间可解类，记为 NP。非确定型图林机解一个问题分两个阶段，即猜测阶段和验证阶段。这里的“求解”并非真正意义上的求解，因为不能保证机器能猜出正确的答案。所谓一个问题能在非确定型图林机上多项式时间可解，意指若机器猜测一个解，它就可以在多项式时间内验证其正确性。

显然，$P\subseteq$NP，因为在确定型图林机上多项式时间可解的任何问题在非确定型图林机上也是多项式时间可解的，此时无需猜测阶段。虽然 NP 中的许多问题似乎比 P 中的问题“难”得多，但还没有人证明 $P\neq$NP。在确定型图林机上，要系统地解 NP 中的某些问题似乎需要指数时间。下面我们给出这类问题的几个例子。

例 3.1.2 背包(knapsack)问题，又称子集和(subset sum)问题。

给定 n 个整数的集合 $A=\{a_1,a_2,\cdots,a_n\}$(通常称 $(a_1,a_2,\cdots,a_n)$ 为背包向量)和一个整数 S，决定是否存在 A 的一个子集 A'，使得 $\sum_{x\in A'}x=S$。

这个问题属于 NP 问题，因为对于给定的子集，易于验证其和是否为 S。然而，找一个子集使其和为 S 要困难得多，因为有 2^n 个可能的子集，试验所有子集的时间复杂性为 T

$=O(2^n)$。

例 3.1.3 可满足性(Satisfiability)问题,简称 SAT 问题

判断一个 n 元布尔(Boolean)函数 $y=f(x_1,x_2,\cdots,x_n)$ 是否存在一组赋值 $(x_1^{(0)},x_2^{(0)},\cdots,x_n^{(0)})$ 使得 $f(x_1^{(0)},x_2^{(0)},\cdots,x_n^{(0)})=1$,即判定 n 元布尔变量组 $x_1,x_2,\cdots,x_n$ 是否存在一组赋值使这些变量的一组给定语句为真。

SAT 问题是一个数理逻辑问题。这个问题也属于 NP 问题,因为对于给定的一组赋值 $(x_1^{(0)},x_2^{(0)},\cdots,x_n^{(0)})$,容易验证 $f(x_1^{(0)},x_2^{(0)},\cdots,x_n^{(0)})$ 是否为 1。然而,找一组赋值使 f 为 1 要困难得多,因为有 2^n 个可能的赋值,试验所有赋值的时间复杂性为 $T=O(2^n)$。

Cook[1]曾证明,可满足性问题具有这样的性质,即 NP 中的每个问题都可以用多项式时间转化成它。这表明,若可满足性问题是易处理的,则 NP 中的每个问题都是易处理的,且若 NP 中的某个问题是难处理的,则可满足性问题也是难处理的。自此以后,人们已证明了许多其它问题[3],例如背包问题也具有上述性质。人们将这类 NP 问题称为 NP 完全的。所谓一个 NP 问题是 NP 完全的,如果 NP 中的任何一个问题都可以通过多项式时间转化为该问题。NP 完全问题的全体记为 NPC. NPC 具有下述性质:若其中的任何一个问题属于 P,则所有的 NP 问题都属于 P,且 $P=$NP。因此,NP 完全问题是 NP 中"最难"的问题。

Co-NP 问题是满足下述性质的问题,问题的补问题是在 NP 中。Co-NP 问题的全体记为 Co-NP。直观地说,NP 中的问题是"决定是否存在一个解"的一类问题,而 Co-NP 中的补问题是"证明不存在解"的一类问题。还不知道是否 NP=Co-NP,但有些问题属于交集 NP∩Co-NP。这类问题的一个例子是"复合数问题":给定整数 n,决定 n 是合数(即存在因子 $p\neq\pm1$ 和 $q\neq\pm1$ 使 $n=pq$)还是素数(即不存在这样的因子)。然而,求因子分解的问题可能比证明它们的存在更难。

需指出的是,在有些应用场合经常说"问题 A 比问题 B 难"或"问题 B 不比问题 A 容易",其含义是问题 B 可通过多项式时间转化为问题 A。

前面所讲的算法和问题的分类主要是根据时间而分的,当然也可以根据空间来分类,这里就不再赘述,感兴趣的读者可参阅文献[2,3]。

最后我们直观地来解释一下密码学中经常使用的概率(或随机)算法。概率(或随机)算法意味着在算法的执行中,在某些状态下作随机选择,即使用一个随机数生成器。概率(或随机)算法有可能失败,但可使失败的概率任意小。

3.2 零知识证明理论

80 年代初,Goldwasser 等人[4,5]提出了零知识证明这一概念。零知识证明是一种协议(非正式地讲,协议就是完成一项任务的一系列步骤,它包括两方或多方),这种协议的一方称为证明者,它试图使被称为验证者的另一方相信某个论断是正确的,却不向验证者提供任何有用的信息。已有大量事实表明零知识证明在密码学中是很有用的。Goldwasser 等人提出的零知识证明是交互式的,也就是证明者和验证者之间必须进行交互,才能实现零知识性,因而称为交互零知识证明。80 年代末,Blum 等人[6,7,8]通过利用一个共同的称作参考串的短随机串代替交互实现了零知识证明这一思想,他们把这种零知识证明称为非

交互零知识证明，这种证明是非交互的、单向的，也就是证明者和验证者在定理证明阶段无需进行交互，就能实现零知识性。非交互零知识证明比交互零知识证明的适用范围更广，因此大大地扩充了零知识证明思想的应用范围。本节主要来论述交互零知识证明和非交互零知识证明的思想及其与密码应用有关的一些理论。

3.2.1 交互零知识证明理论

在交互零知识证明(Interactive zero knowledge proofs)的研究中，目前人们最关心的基本模型有两种。一种是GMR模型[4]，在这种模型中，证明者具有无限的计算能力，验证者具有多项式时间的计算能力，证明指的是语言成员问题，即输入I是否是语言L的一个成员。GMR的零知识证明不是真正的零知识证明，这是因为在证明中，证明者向验证者揭露了知识的1比特，即$I\in L$。但除此之外，再没有其它任何附加的信息泄露给验证者，通常称这种交互零知识证明为成员或定理的零知识证明(Zero knowledge proofs of membership or theorem)。另一种是FFS模型[9]，在这种模型中，证明者和验证者均具有多项式时间的计算能力，证明者的目的不是向验证者证明$I\in L$，而是证明他知道I关于L的状况。FFS的零知识证明是真正的零知识证明，因为在证明中，验证者没有得到任何信息，他连$I\in L$还是$I\notin L$都不知道，但他相信这个证明，通常称这种交互零知识证明为知识或身份的零知识证明(Zero knowledge proofs of knowledge or identity)。

我们用一个例子来说明一下这两种交互零知识证明的差别。例如，若一个数学家解决了Fermat大定理，当他使用GMR的零知识证明时，他不仅能使验证者相信他解决了这个问题，而且能使验证者知道他是证明了这个问题还是否定了这个问题。当他使用FFS的零知识证明时，他只能使验证者相信他解决了这个问题，但验证者并不知道他证明了这个问题还是否定了这个问题。当然，这仅仅是一个例子，我们知道Fermat大定理已被证明是正确的。

为了便于理解零知识，我们引用文献[10]中关于洞穴的故事来解释这一概念。这个洞穴如图3.2.1所示，里面有一个秘密，知道咒语的那些人能打开C和D之间的秘密之门。对其他任何人来说，这两条路都是死胡同。

证明者P知道这个洞穴的秘密。他想让验证者V相信这一事实，但他不想泄露咒语。下面是P怎样使V相信的过程：

(1)V站在A点；

(2)P走进洞穴，到达C点或D点；

(3)在P消失在洞穴中之后，V走到B点；

(4)V向P喊，叫P或者；

(a)从左通道出来，或者(b)从右通道出来；

(5)P答应了，如果有必要的话他就用咒语打开秘密之门；

(6)P和V重复步骤(1)至(5)。

因为P没有办法重复猜出V要他从

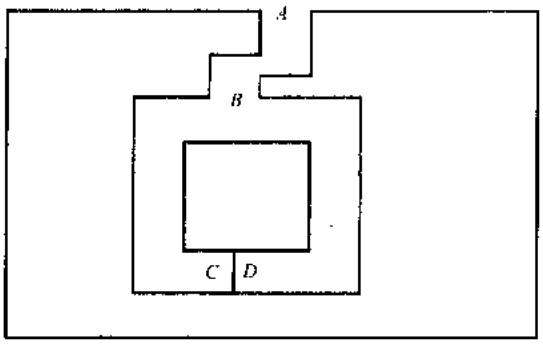


图3.2.1 零知识洞穴

哪一边出来，所以如果 P 不知道这个秘密，那么他只能从进去的一边出来而不能从另一边出来。在每一轮中 P 有 1/2 的机会猜中 V 会叫他从哪一边出来，所以有 1/2 的机会愚弄 V。在两轮中 P 愚弄 V 的机会是 1/4。而在 n 轮后 P 愚弄 V 的机会是 $1/2^n$。当 $n=16$ 时，P 每次只有 1/65536 的机会愚弄 V。因此，如果所有 16 次 P 的证明都是对的，那么 V 可以相信 P 一定知道开启 C 点和 D 点间门的咒语。

3.2.1.1 成员的零知识证明

为了介绍成员的零知识证明这一概念，我们首先必须引入一系列其它概念。

成员的交互证明系统

一个交互图林机(interactive Turing machine)是一个具有一条只读输入带、一条工作带、一条随机带、一条只读通信带和一条只写通信带的图林机。随机带上包含一条无限长的随机比特序列，它只能从左到右地读入。我们说一个交互图林机掷一个硬币意指它从自己的随机带上读下一个比特。

一个交互协议(interactive protocol)是满足下列两个条件的一对有序图林机(A,B)：(1)A 和 B 共享同一条输入带；(2) B 的只写通信带是 A 的只读通信带，反之，B 的只读通信带是 A 的只写通信带。机器 A 具有无限的计算能力，而机器 B 具有多项式时间的计算能力。所谓一个机器具有多项式时间的计算能力是指该机器关于任何输入所完成的全部计算或操作所需的时间都是它的输入的长度的一个多项式。一个交互协议如图 3.2.2 所示，它的工作原理是：B 首先开始工作，然后两台机器轮流工作。在机器 A(或 B)的工作阶段，A(或 B)首先使用公共输入带和它的工作带、通信带、随机带上的内容完成某种内部计算；其次，A(或 B)在它的只写通信带上为 B(或 A)写一个串。A(或 B)的第 i 条消息是 A(或 B)在它的第 i 个工作阶段写在它的通信带上的全部串。一旦机器 A(或 B)写了它的消息，它就不工作，而机器 B(或 A)开始工作(除非协议被终止)。无论哪一台机器都能通过在一个工作阶段不发送任何消息来终止协议的计算。机器 B 通过输出接收(或拒绝)和终止协议来接收(或拒绝)公共输入。机器 B 的计算时间是指 B 在各个工作阶段的计算时间的总和，这个时间是多项式的。

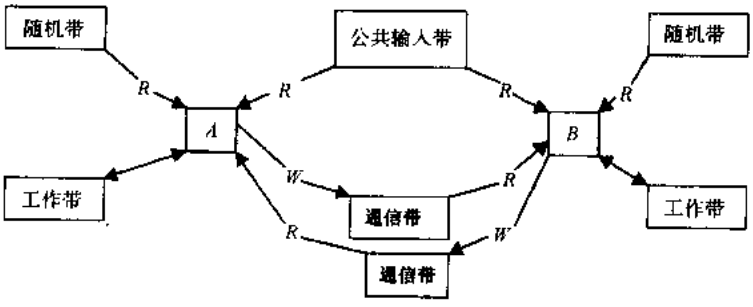


图 3.2.2 一个交互协议

"←→"表示一个读/写头，"←R−"表示一个只读头，"−W→"表示一个只写头

我们所说的"语言"是指一个集合 L。因为通常集合 L 中的每个元素 x 都可编码成一个 0、1 有限长串，该串的长度称为 x 的长度，记为 $|x|$，所以可以抽象地将 L 视作集合 $\{0,1\}^*$ 的一个子集。这里 $\{0,1\}^*$ 表示所有有限长的 0、1 串构成的集合。

设 $L\subset\{0,1\}^*$ 是一个语言，(A,B) 是一个交互协议。我们说 (A,B) 对 L 是一个成员的交互证明系统(interactive proof system of membership)，如果它满足下列两个条件：

(1)完全性(Completeness)：对每一个 $k>0$ 和充分长的 $x\in L$，将 x 作为 (A,B) 的输入，B 终止协议并至少以 $1-|x|^{-k}$ 的概率接收 x。这里的概率是相对于协议 (A,B) 所有可能的掷硬币而言的。

(2)合理性(Soundness)：对每一个 $k>0$ 和充分长的 $x\notin L$，对任意的交互图林机 A'，将 x 作为 (A',B) 的输入，B 至多以 $|x|^{-k}$ 的概率接收 x。这里的概率是相对于协议 (A',B) 所有可能的掷硬币而言的。

注意，在上述定义中，误差率 $|x|^{-k}$ 可以减弱为某常数 $0\leqslant\varepsilon<1/2$，也可加强为 $2^{-|x|}$，这三种误差率的交互证明系统是等价的[11]。

条件(1)本质上是说，如果 $x\in L$，A 能以很大的概率说服 B 接收 x。条件(2)是说，如果 $x\notin L$，无论 A' 采取任何欺骗策略，A' 说服 B 接收 x 的概率很小。事实上，B 无需相信正在与它进行交互的机器是否诚实，只要相信它自己掷硬币的随机性就足够了。条件(2)也表明，交互证明系统的定义依赖于 B，而根本不依赖于 A。在一个交互证明系统 (A,B) 中，通常将 A 称为证明者，B 称为验证者。

下面我们将对二次剩余(quadratic residues)问题描述一个成员的交互证明系统。所谓二次剩余问题是指给定 $n\in N$(N 是全体自然数之集)和 $y\in Z_n^*=\{y\in Z_n|\gcd(y,N)=1\}$，确定 y 是否是一个模 n 的二次剩余。关于二次剩余的定义及其基本性质参见附录。记

$$\mathrm{QR}=\{(n,y)|n\in N,y\in Z_n^*,y\text{ 是模 }n\text{ 的一个二次剩余}\}$$

$$\mathrm{QNR}=\{(n,y)|n\in N,y\in Z_n^*,y\text{ 不是模 }n\text{ 的一个二次剩余，但}(y/n)=1\}$$

其中 (y/n) 表示 y 模 n 的 Jacobi 符号，n 和 y 以二进制的形式给出。通常将 QR 称为二次剩余语言，QNR 称为二次非剩余(quadratic non-residues)语言。下面我们就对二次非剩余语言 QNR 来描述一个成员的交互证明系统。

设 (A,B) 是一个交互协议，(n,y) 是它们的公共输入，$m=|n|$，其中 $|n|$ 表示 n 的二进制表示的长度。B 检查是否 $n\geqslant 1$，$y\in Z_n^*$，$(y/n)=1$，如果不是，停机，否则，A 和 B 重复执行下列步骤 m 次：

(1)B 从它的随机带上读出一个整数 $i=0$ 或 1(将该整数称作口令(challenge))和一个整数 $r\in Z_n^*$，即 B 随机地选择一个整数 $i=0$ 或 1 和一个整数 $r\in Z_n^*$，计算 $w=y^ir^2\bmod n$，并将 w 写在它的只写通信带上，即将 w 发送给 A；

(2)A 从它的只读通信带上读出 w，即 A 从 B 处收到 w 后，它验证是否 $(n,w)\in\mathrm{QR}$，如果 $(n,w)\in\mathrm{QR}$，那么 A 定义 $j=0$，否则它定义 $j=1$，然后它将 j 写在只写通信带上，即将 j 发送给 B(通常将 j 称作对口令的回答(response))；

(3)B 验证是否 $i=j$。

如果在 m 轮中的每一轮都有 $i=j$，那么 B 接收 A 的证明，即认为 $(n,y)\in\mathrm{QNR}$。

上述交互协议是一个交互证明系统。这是因为，如果 $(n,y)\in\mathrm{QNR}$，由 A 和 B 的执行过程可知，在每一轮中都有 $i=j$，B 以概率 1 接收 A 的证明，即该协议是完全的。如果 $(n,y)\notin\mathrm{QNR}$，则可假定 $(n,y)\in\mathrm{QR}$，这是因为其它情况由 B 在开始时的检查可被排除或拒绝。此时无论 $i=0$ 还是 $i=1$，w 都是模 n 的二次剩余，A 无法确定是 $j=0$ 还是 $j=1$，它猜中的概率为 1/2，这样只有当 m 轮中的每一轮中的 j 都猜中时，B 才接收，这种做法成功

的概率只有 $1/2^m$。因此，当 $(n,y)\notin \mathrm{QNR}$ 时，B 接收 A' 的证明的概率不超过 $1/2^m$，即该协议是合理的。

我们来看一下 B 的计算时间。数论知识告诉我们，对任意给定的 $n\in N$ 和 $y\in Z_n$，利用 Euclidean 算法可在 $|n|$ 的多项式时间内计算出是否 $y\in Z_n^*$，而且 (y/n) 也可在 $|n|$ 的多项式时间内计算出。而在 B 的每一轮中，B 计算 $w=y^jr^2 \bmod n$ 和验证 $i=j$ 的时间都是多项式的，故 B 的计算时间是多项式的。

随机变量的不可区分性和可遍近性

设 $L\subset\{0,1\}^*$，$U=\{U_{(x)}\}_{x\in L}$ 和 $V=\{V_{(x)}\}_{x\in L}$ 是两族随机变量，所有随机变量都在 $\{0,1\}^*$ 中取值。我们想表达这样一个事实：随着 x 的长度的增加，$U_{(x)}$ 本质上可由 $V_{(x)}$ 来“取代”。现在我们考虑下述框架：

从 $U_{(x)}$ 或者从 $V_{(x)}$ 中抽取出一批随机样本并将这些随机样本交给一个判决者。判决者在研究这些样本之后，他将作出判决。如果样本来自 $U_{(x)}$，则判定为 0，如果样本来自 $V_{(x)}$，则判定为 1。如果随着 x 的长度的增加，任何判决者都无法作出判决，或者只能与 $U_{(x)}$ 和 $V_{(x)}$ 无关地胡乱随意地判决，那么我们就说 $U_{(x)}$ 本质上可由 $V_{(x)}$ 来“取代”，或者说 $U_{(x)}$ 和 $V_{(x)}$ 不可区分。在这个框架中有两个相关参数值得考虑，即样本的数目和判决者作出判决所需的时间。通过对这两个参数作不同的限制就会得到不同的随机变量不可区分的概念，目前最关心的不可区分的概念有三个，即相等(equality)、统计不可区分(Statistical indistinguishability)和计算不可区分(computational indistinguishablility)。下面我们来分别介绍这三个概念。

我们说两族随机变量 $\{U_{(x)}\}_{x\in L}$ 和 $\{V_{(x)}\}_{x\in L}$ 在语言 L 上是相等的，如果对每个充分长的 $x\in L$，$U_{(x)}$ 和 $V_{(x)}$ 的概率分布相等，即对每个 $\alpha\in\{0,1\}^*$，$P(U_{(x)}=\alpha)=P(V_{(x)}=\alpha)$。这时也称 $\{U_{(x)}\}_{x\in L}$ 和 $\{V_{(x)}\}_{x\in L}$ 完全不可区分(perfect indistinguishability)。也就是说，这两个带参数 x 的随机变量族，对充分长的 $x\in L$，是相等的。

由定义可以看出，如果两族随机变量 $\{U_{(x)}\}_{x\in L}$ 和 $\{V_{(x)}\}_{x\in L}$ 在语言 L 上是相等的，那么对充分长的 $x\in L$，判决者即使具有无限的计算能力和拥有无穷多的样本也无法判定这些样本来自 $U_{(x)}$ 还是来自 $V_{(x)}$。

我们说两族随机变量 $\{U_{(x)}\}_{x\in L}$ 和 $\{V_{(x)}\}_{x\in L}$ 在语言 L 上是统计不可区分的，如果对任意常数 $c>0$ 和每个充分长的 $x\in L$，有

$$\sum_{\alpha\in\{0,1\}^*}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)|<|x|^{-c}$$

由定义可以看出，如果两族随机变量 $\{U_{(x)}\}_{x\in L}$ 和 $\{V_{(x)}\}_{x\in L}$ 在语言 L 上是统计不可区分的，那么对充分长的 $x\in L$，对拥有多项式个样本和具有无限计算能力的判决者，他也基本上无法判定这些样本来自 $U_{(x)}$ 还是 $V_{(x)}$。

例 3.2.1 设 $U_{(x)}$ 对所有长度为 $|x|$ 的串赋予相同的概率 $2^{-|x|}$，$V_{(x)}$ 对除了全 0 和全 1 的长度为 $|x|$ 的串赋予相同的概率 $2^{-|x|}$，对长度为 $|x|$ 的全 0 串和全 1 串分别赋予概率 0 和 $2^{-|x|+1}$，则 $\{U_{(x)}\}_{x\in L}$ 和 $\{V_{(x)}\}_{x\in L}$ 是语言 $L=\{0,1\}^*$ 上统计不可区分的两族随机变量。

上面我们在给出完全不可区分和统计不可区分的定义时，实际上将判决者视作一个概率图林机，即带有一条随机带的图林机。按上述两个定义，我们在定义计算不可区分性时，应该将判决者视作一个多项式时间的概率图林机，但我们不这样做，而是将判决者视

作一个多项式规模的电路族，这是因为通常认为这种电路族是一种可能比多项式时间的概率图林机接收能力更强的计算装置。详细理由可参阅文献[5]。

设 $C=\{C_x\}_{x\in L}$是一族布尔电路，C_x 是输出仅为 0 或 1 的布尔电路，C_x 的输入是以 x 为参数的随机变量，即 C_x 的输入是按参数 x 确定的随机变量分布的随机串。如果存在一个常数 $e>0$，使得对所有的布尔电路 $C_x\in C$ 至多有 $|x|^e$ 个门（门包括与门、或门、非门等），我们称 C 为多项式规模的电路族。

为了把来自某一概率分布的样本输入多项式规模的电路，我们将只考虑多项式界随机变量族。所谓 $U=\{U_{(x)}\}_{x\in L}$是一个多项式界随机变量族，意指存在一个常数 $d>0$，使得对所有的随机变量 $U_{(x)}\in U$ 只对长度不超过 $|x|^d$ 的串分配正概率。

设 $U=\{U_{(x)}\}_{x\in L}$和 $V=\{V_{(x)}\}_{x\in L}$是两个多项式界随机变量族，$C=\{C_x\}_{x\in L}$是多项式规模的电路族，我们用 $P(U,C,x)$表示按 $U_{(x)}$分布的随机串作为输入，C_x 输出 1 的概率。我们说 U 和 V 在语言 L 上是计算不可区分的，如果对任意常数 $c>0$ 和每个充分长的 $x\in L$，有 $|P(U,C,x)-P(V,C,x)|<|x|^{-c}$。

由定义易知，对于两个随机变量族 $U=\{U_{(x)}\}_{x\in L}$和 $V=\{V_{(x)}\}_{x\in L}$，如果它们在语言 L 上是完全不可区分的，那么它们在语言 L 上必定是统计不可区分的。我们现在来说明，对于两个多项式界随机变量族 $U=\{U_{(x)}\}_{x\in L}$和 $V=\{V_{(x)}\}_{x\in L}$，如果它们在语言 L 上是统计不可区分的，那么它们在语言 L 上必定是计算不可区分的。

设 C_x 是一个电路，S_x 是使得 C_x 的输出为 1 的输入之集。因为 U 和 V 是统计不可区分的，所以对任意常数 $c>0$ 和每个充分长的 $x\in L$，有

$$\sum_{\alpha\in\{0,1\}^*}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)|<|x|^{-c}$$

而

$$|P(U_{(x)}\in S_x)-P(V_{(x)}\in S_x)|=\Big|\sum_{\alpha\in S_x}P(U_{(x)}=\alpha)-\sum_{\alpha\in S_x}P(V_{(x)}=\alpha)\Big|$$

$$\leq\sum_{\alpha\in S_x}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)\Big|\leq\sum_{\alpha\in\{0,1\}^*}|P(U_{(x)}=\alpha)-P(V_{(x)}=\alpha)\Big|$$

所以

$$|P(U_{(x)}\in S_x)-P(V_{(x)}\in S_x)|<|x|^{-c}$$

又

$$P(U,C,x)=P(U_{(x)}\in S_x)$$
$$P(V,C,x)=P(V_{(x)}\in S_x)$$

所以

$$|P(U,C,x)-P(V,C,x)|<|x|^{-c}$$

故 U 和 V 在语言 L 上是计算不可区分的。

现在我们来定义随机变量的可逼近性(approximability)。设 M 是一个关于输入 x 以概率 1 停机的概率图林机，我们用 $M(x)$来表示一个随机变量，该随机变量的概率分布为：对每一个串 α，$P(M(x)=\alpha)=p(M$ 关于输入 x 输出 $\alpha)$，即 $P(M(x)=\alpha)$定义为 M 关于输入 x 输出 α 的概率。

设 $L\subseteq\{0,1\}^*$，$U=\{U(x)\}_{x\in L}$是一族随机变量，我们说 U 在语言 L 上是完全(统计、计算)可逼近的，如果存在一个多项式时间的概率图林机 M，使得$\{M(x)\}_{x\in L}$和$\{U_{(x)}\}_{x\in L}$

在 L 上是完全(统计、计算)不可区分的。

由定义可知,随机变量的可逼近性和随机变量的不可区分性密切相关,每一种不可区分性对应一种可逼近性。如果 U 在 L 上是完全逼近的,那么 U 在 L 上必是计算可逼近的。当然我们在谈论随机变量的计算不可区分性和计算可逼近性时,是指多项式界随机变量的计算不可区分性和计算可逼近性。

成员的零知识证明

非正式地讲,一个成员的交互证明系统(A,B)是由 A 和 B 两方所执行的一个协议。A 试图通过执行协议,说服验证者 B 相信某个定理 T(如 $x\in L$)是正确的。如果 T 为假(如 $x\notin L$),则即使 A 不遵循协议,采取任何欺骗策略,也无法说服 B 相信 T 为真。但在一个成员的交互证明系统中,如果一个有欺骗行为的验证者 B' 不遵循协议,他妄图从 A 那里得到除了 T 为真以外的其它信息,即 B' 通过协议可能获得了他不应得到的知识。我们现在来考虑即使 B' 不遵循协议,采取任何欺骗策略,也无法从 A 那里得到除了 T 为真以外的任何其它信息的交互协议和交互证明系统。

首先我们描述一个欺骗验证者(cheating verifier) B',它预先拥有某些额外信息且不遵循协议。

设(A,B)是一个交互协议,B' 是一个带有一条额外输入带的交互图林机。当公共输入为 x 时,B' 的输入为(x,H),H 是 B' 的额外输入,且 H 的长度不超过$|x|$的多项式。当 B' 与 A 交互时,A 只看到它的输入带上的 x,而 B' 看到的是(x,H),此时假定 B' 的所有计算时间不超过$|x|$的多项式。H 的一个最好的解释是把它解释为欺骗验证者 B' 已经拥有的关于 x 的一些知识或欺骗验证者在执行协议(A,B')之前极力使用从 A 得到的知识执行其它协议所获得的交互历史。

对关于公共输入为 x 和额外输入为 H 的一段协议,我们定义 B' 的观察(View)是 B' 看到的一切事情。设 σ 和 ρ 分别是包含在 A 和 B' 的随机带中的两个串,不妨设 A 和 B' 关于这些随机选择的计算由 n 轮构成,并且 B' 首先开始工作,这里 a_i 和 b_i 分别是 A 和 B' 的第 i 个消息。那么我们说$(\rho,b_1,a_1,\cdots,b_n,a_n)$是 B' 关于输入(x,H)的观察。设 $\mathrm{View}_{A,B'}(x,H)$是一个随机变量,其值是这个观察。为方便起见,我们把每个观察视作来自$\{0,1\}^*$的长度不超过$|x|$的多项式的一个串。注意,我们可以把 B' 的工作带中的内容包括在 $\mathrm{View}_{A,B'}(x,H)$中,也可以排除在 $\mathrm{View}_{A,B'}(x,H)$之外,这都无本质差别,因为它们都能在多项式时间内从 $x,H,\rho,a_1,a_2,\cdots,a_n$ 求得。

设 $L\subseteq\{0,1\}^*$是一个语言,(A,B)是一个交互协议,B' 是如上所述的欺骗验证者。我们说(A,B)对 B' 关于语言 L 是完全(统计、计算)零知识的,如果随机变量族 $\mathrm{View}_{A,B'}=\{\mathrm{View}_{A,B'}(x,H)\}_{(x,H')\in L'}$关于语言 $L'=\{(x,H)\mid x\in L,|H|$不超过$|x|$的多项式$\}$是完全(统计、计算)可逼近的。我们说$(A,B)$关于语言 L 是完全(统计、计算)零知识的,如果它是对所有的多项式时间的概率交互图林机 B' 关于语言 L 是完全(统计、计算)零知识的。

由定义可知,如果(A,B)关于语言 L 是完全(统计、计算)零知识的,那么所有的多项式时间的概率交互图林机,无论采用何种欺骗措施,都无法通过和 A 交互抽取关于 L 的成员的一些附加信息。这表明该定义只依赖于 A,而根本不依赖于 B。

零知识的定义中考虑了两种概率分布:(1)一种是由一个概率多项式时间验证者 B' 在与证明者 A 交互之后产生的概率分布;(2)另一种是由一个概率多项式时间机器 M 关

于输入产生的概率分布，M 没有和任何证明者交互。零知识意味着对每个类型(1)的分布存在一个类型(2)的分布，使得这两个分布是“本质上相等的”。直观地说，零知识意味着 B 与 A 交互所得到的信息是 B 单独也能得到的。有时将(2)中的 M 称为伪造算法(forging algorithm)或模拟器(simulator)。

下面我们来定义成员的零知识证明系统这一概念。

设 $L\subset\{0,1\}^*$ 是一个语言，(A,B) 是一个交互协议，我们说 (A,B) 对语言 L 是一个成员的完全(统计、计算)零知识证明系统，如果它对 L 是一个成员的交互证明系统并且关于 L 是一个完全(统计、计算)零知识协议。

由定义易知，如果 (A,B) 对语言 L 是一个成员的完全零知识证明系统，那么 (A,B) 对语言 L 一定是一个成员的统计零知识证明系统。如果 (A,B) 对语言 L 是一个成员的统计零知识证明系统，那么 (A,B) 对语言 L 一定是一个成员的计算零知识证明系统。

通常将计算零知识证明系统称为零知识证明系统，简称零知识证明。

如果 (A,B) 对 L 是一个成员的零知识证明系统，那么所有的概率多项式时间交互图林机，无论采用何种欺骗措施，都不可能通过和 A 交互抽取除了 $x\in L$ 之外的别的信息。

前面我们已对二次非剩余语言 QNR 描述了一个成员的交互证明系统，现在我们来考察一下该系统的零知识性。

假定有一个知道 A 和 B 的公共输入 $x=(n,y)\in$QNR 的第三方 C，C 随机地选择 $\tilde{r}\in Z_n^*$ 和 $\tilde{i}\in\{0,1\}$，并计算 $\tilde{w}=y^{\tilde{i}}\tilde{r}^2\bmod n$。$C$ 将 $\tilde{w}$ 给 B，但 B 并不知道 $\tilde{w}$ 是否是一个二次剩余。B 在第(2)步将 $\tilde{w}$ 发送给 A，A 在第(3)步对 B 作出回答。这样 B 就借助 A 能够回答他事先不知道的并且有可能不能解决的问题。直观上看，该系统不是零知识的，这里不再作详细讨论，文献[5]中证明了该系统是统计零知识证明系统。

本节最后我们对二次剩余语言 QR 描述一个成员的零知识证明系统。

设 (A,B) 是一个交互协议，(n,y) 是它们的公共输入，$m=|n|$，其中 $|n|$ 表示 n 的二进制表示的长度。B 检查是否 $x\geqslant 1$，$y\in Z_n^*$，如果不是，停机，否则，A 和 B 重复执行下列步骤 m 次：

(1)A 从它的随机带上读出一个整数 $r\in Z_n^*$，即 A 随机地选择一个整数 $r\in Z_n^*$，计算 $w=r^2\bmod n$，并将 w 写在它的只写通信带上，即将 w 发送给 B；

(2)B 从它的随机带上读出一个整数 $i=0$ 或 1，即 B 随机地选择一个整数 $i=0$ 或 1，并将 i 写在它的只写通信带上，即将 i 发送给 A；

(3)A 从它的只读通信带上读出 i，即 A 从 B 处收到 i 后，它计算 $Z=u^ir\bmod n$，这里 u 是 y 的一个平方根，即 $y=u^2\bmod n$，并将 Z 写在它的只写通信带上，即将 Z 发送给 B；

(4)B 从它的只读通信带上读出 Z，即 B 从 A 处收到 Z 后，它验证是否 $Z^2\equiv y^iw\bmod n$。

如果在 m 轮中的每一轮都有 $Z^2\equiv y^iw\bmod n$，那么 B 接收 A 的证明，即认为 $(n,y)\in$QR。

类似于 QNR 的情形，容易说明 B 的所有计算时间都不超过 $|n|$ 的多项式。

下面我们来说明上述系统对语言 QR 是一个成员的完全零知识证明系统。

完全性：如果 $(n,y)\in$QR，由 A 和 B 的执行过程可知，在每一轮中都有 $Z^2\equiv y^iw\bmod n$，所以 B 以概率 1 接收 A 的证明。

合理性：如果$(n,y)\notin QR$，则可假定$n\geqslant 1$，$y\in Z_n^*$，y不是模n的一个二次剩余，这是因为其它情况由B在开始时的检查可被排除或拒绝。在每一轮中，B从A'处收到的w不可能使w和yw都有模n的平方根。因为A'事先没有看到B随机选择的口令i，他只好去猜，猜中的概率为1/2，所以在每一轮中，B接收A'的证明的概率为1/2，这样只有当A'在m轮中的每一轮中都能正确地猜中口令i时，B才接收A'的证明，这种做法成功的概率只有$1/2^m$。因此B至多以概率$1/2^m$接收A'的证明。

上述两个性质表明，协议(A,B)对QR是一个成员的交互证明系统。下面我们来说明该协议是(完全)零知识的。

(完全)**零知识性**：设B'是和A交互的任一个多项式时间交互图林机，$(n,y)\in QR$是(A,B')的公共输入，$m=|n|$，$|n|$表示n的二进制表示的长度，H是B'的额外输入，H也可能是一个空串。B'关于输入$((n,y)H)$的观察值为$((w_1,i_1,Z_1);(w_2,i_2,Z_2);\cdots;(w_m,i_m,Z_m))$，其中$(w_j,i_j,Z_j)(1\leqslant j\leqslant m)$是$B'$关于输入$((n,y),H)$在第$j$轮的观察值，$w_j$是$A$随机选择的一个模$n$的二次剩余，$i_j$是$B'$随机选择的一个整数(0或1)，$Z_j$是$A$发送给$B'$的$y^{i_j}w_j$模$n$的一个平方根。设$\mathrm{View}_{A,B'}((n,y),H)=((\boldsymbol{w}_1,\boldsymbol{i}_1,\boldsymbol{Z}_1);(\boldsymbol{w}_2,\boldsymbol{i}_2,\boldsymbol{Z}_2);\cdots;(\boldsymbol{w}_m,\boldsymbol{i}_m,\boldsymbol{Z}_m))$是一个随机变量，其值是$B'$关于输入$((n,y),H)$的观察值。$B'$可能遵循协议，也可能不遵循协议。

为了证明上述交互证明系统是完全零知识的，我们需对每一个B'(诚实的或不诚实的验证者)，构造出一个相应的概率多项式时间图林机$M_{B'}$，使得$M_{B'}((n,y),H)=\mathrm{View}_{A,B'}((n,y),H)$。$M_{B'}$扮演$A$的角色，并将$B'$用作一个可重新起动的子程序。$M_{B'}$极力猜测$B'$将在第$j$轮中选择的$i_j$。也就是说，在第$i$轮中$M_{B'}$首先产生一个形式为$(w_j,i_j,Z_j)$的随机的三元组，其中$Z_j^2\equiv y^{i_j}w_j \bmod n$，然后运行机器$B'$，看$B'$选择的口令$i_j'$。如果$M_{B'}$的猜测$i_j$和$B'$产生的$i_j'$相等，那么将$(w_j,i_j,Z_j)$级联到伪造的观察之后。否则，将这个三元组删掉，$M_{B'}$猜测一个新的口令$i_j$并将$B'$的状态调回原状态重新起动$B'$。这里的"状态"是指由机器使用的所有变量的值。比方说，令$\boldsymbol{V}_j=((\boldsymbol{w}_1,\boldsymbol{i}_1,\boldsymbol{Z}_1);(\boldsymbol{w}_2,\boldsymbol{i}_2,\boldsymbol{Z}_2);\cdots;(\boldsymbol{w}_j,\boldsymbol{i}_j,\boldsymbol{Z}_j))$是一个随机变量，在$B'$执行第$j+1$轮时，$\boldsymbol{V}_j$已取定一个值$v_j$，将$v_j$称作$B'$在执行$j+1$轮时的一个状态，简称状态，记为$S_j(B')$。约定$S_0(B')=$空串。

对B'，现在我们来详细描述一个多项式时间的概率图林机$M_{B'}$，$M_{B'}$的输入为$(n,y)\in QR$和串。用T表示由$M_{B'}$伪造的观察，置$T=$空串。$M_{B'}$对$j=1\sim m$执行下列步骤：

(1)$S_{j-1}(B')=T$；

(2)$M_{B'}$从它的随机带上读出一个整数$i_j=0$或1和一个整数$Z_j\in Z_n^*$，即$M_{B'}$随机选择一个整数$i_j=0$或1和一个整数$Z_j\in Z_n^*$，并计算$w_j=Z_j^2y^{-i_j}\bmod n$；

(3)调用B'的第j轮，B'的前$j-1$轮的观察已由(1)中赋值。给B'输入w_j，获得一个整数i_j'，如果$i_j=i_j'$，那么将(w_j,i_j,Z_j)级联到T的尾部，否则$M_{B'}$返回步骤(2)，直到$i_j=i_j'$为止。

我们先来看一下$M_{B'}$的计算时间。$M_{B'}$随机选择$Z\in Z_n^*$的方法如下：$M_{B'}$首先随机选择一个长度为m的比特串，检测看它是否属于Z_n^*，检测Z是否属于Z_n^*可在$m=|n|$的多项式时间内完成；如果不是，$M_{B'}$随机选择一个长度为$m-1$的比特串，检测看它是否属于Z_n^*；依次类推，直到$M_{B'}$随机选择一个长度为1的比特串进行检测为止。平均来说，这m

个串中有 Z_n^* 中的元素，故平均来说 M_B 随机选择 $Z\in Z_n^*$ 可在 $m=|n|$ 的多项式时间内完成。

我们再看一下猜到 $i_j=i_j'$ 的平均运行时间有多长。不管 B' 怎样产生它的口令 i_j'，M_B 的猜测 i_j 和 i_j' 一样的概率是 1/2。因此，平均来说，对每一个级联到伪造的观察尾部的三元组，M_B 将产生两个三元组。故平均运行时间是 $m=|n|$ 的多项式。可见，M_B 是一个概率多项式时间图林机。

下面我们用数学归纳法来证明 $M_B((n,y),H)=\mathrm{View}_{A,B}((n,y),H)$。对每一个 $j:0\leqslant j\leqslant m$，令 T_j 表示第 j 轮末的部分观察 $((w_1,i_1,Z_1);(w_2,i_2,Z_2);\cdots;(w_j,i_j,Z_j))$ 的全体之集，设 M_B 在 T_j 上的概率分布为 $P_{M_B,j}$，$\mathrm{View}_{A,B}$ 在 T_j 上的概率分布为 $P_{\mathrm{View}_{A,B},j}$。注意到 $P_{M_B,m}=P_{M_B}$，$P_{\mathrm{View}_{A,B},m}=P_{\mathrm{View}_{A,B}}$，因此如果我们能证明对所有的 $j:0\leqslant j\leqslant m$，在 T_j 上的两个概率分布 $P_{M_B,j}$ 和 $P_{\mathrm{View}_{A,B},j}$ 是相等的，那么我们就证明了

$$M_B((n,y),H)=\mathrm{View}_{A,B}((n,y),H)$$

$j=0$ 相对应的是机器的开始，此时 T_0 是空集，因此在 T_0 上的两个概率分布显然是相等的。

假定对某一 $j\geqslant 1$，在 T_{j-1} 上的两个概率分布 $P_{M_B,j-1}$ 和 $P_{\mathrm{View}_{A,B},j-1}$ 是相等的，我们现在证明在 T_j 上的两个概率分布 $P_{M_B,j}$ 和 $P_{\mathrm{View}_{A,B},j}$ 是相等的。

考虑交互证明中的第 j 轮。B' 随机选择整数 $i_j=0$ 的概率是某一实数 P_1，因此，随机选择整数 $i_j=1$ 的概率是 $1-P_1$，这里 P_1 依赖于机器 B' 的第 j 轮的状态。在交互证明中，我们注意到，每一个 w 都由证明者 A 等可能地选择，又因为无论对哪一个可能的口令 i_j，所有的平方根都是等可能的，所以每一个 Z 都独立于 P_1 等可能地出现。因此，观察的第 j 个三元组是 (w,i,z) 的概率是 $P_1/t(i=0$ 时$)$ 或 $(1-P_1)/t(i=1$ 时$)$，这里 t 是 Z_n^* 中平方根的个数。

我们对 M_B 作一个类似的分析。在 M_B 的每一轮中的每一次迭代中，M_B 以概率 $1/t$ 选择任何平方根 Z。$i=0$ 并且 B' 的口令 $i'=0$ 的概率是 $P_1/2$；$i=1$ 并且 B' 的口令 $i'=1$ 的概率是 $(1-P_1)/2$。在这些情况中的每一种情况，(w,i,Z) 都能被作为伪造的观察的第 j 个三元组。在每一轮中的每一次迭代中，以 1/2 的概率在带上没写任何东西。对 $i=0$ 的情形，在第 j 轮中，一个三元组 $(w,0,Z)$ 在 l 次迭代后被作为伪造的观察的第 j 个的概率是 $P_1/(2^l\times t)$。因此，$(w,0,Z)$ 是伪造的观察的第 j 个三元组的概率是 $P_1/(2\times t)+P_1/(2^2\times t)+P_1/(2^3\times t)+\cdots=P_1/(2\times t)\left(1+\frac{1}{2}+\frac{1}{4}+\cdots\right)=P_1/t$。类似于 $i=0$ 的情形对 $i=1$ 的情形进行分析可得，$(w,1,Z)$ 是伪造的观察的第 j 个三元组的概率是 $(1-P_1)/t$。因此，在第 j 轮末，在 T_j 上的两个概率分布分别为 $P_{M_B,j}=P_{M_B,j-1}\times P_1/t(i=0$ 时$)$ 或 $P_{M_B,j-1}\times(1-P_1)/t(i=1$ 时$)$ 和 $P_{\mathrm{View}_{A,B},j}=P_{\mathrm{View}_{A,B},j-1}\times P_1/t(i=0$ 时$)$ 或 $P_{\mathrm{View}_{A,B},j-1}\times(1-P_1)/t(i=1$ 时$)$，由归纳假设知，$P_{\mathrm{View}_{A,B},j-1}=P_{M_B,j-1}$，故在 T_j 上的两个概率分布是相等的。由归纳法可知，两个概率分布 P_{M_B} 和 $P_{\mathrm{View}_{A,B}}$ 是相等的。

我们现在针对子群成员(subgroup membership)问题给出另一个成员的完全零知识证明系统的例子。所谓子群成员问题是指给定 $n,l\in N$ 和 $\alpha,\beta\in Z_n^*$，$\alpha\neq\beta$，α 在 Z_n^* 中的阶为 l，则 $\langle\alpha\rangle=\{1,\alpha,\alpha^2,\cdots,\alpha^{l-1}\}$ 是由 α 生成的 Z_n^* 的一个 l 阶子群，确定 β 是否属于 $\langle\alpha\rangle$。记 $\mathrm{SM}=\{(n,l,\alpha,\beta)\mid n,l\in N,\alpha,\beta\in Z_n^*,\alpha$ 的阶是 $l,\alpha\neq\beta,\beta\in\langle\alpha\rangle\}$，$n,l,\alpha$ 和 β 都以二进制的

形式给出。将 SM 称为子群成员语言。下面我们针对 SM 语言来描述一个成员的完全零知识证明系统，这个协议的分析与对 QR 语言的完全零知识证明系统的分析类似，这里不在详细分析，感兴趣的读者可自己给出分析过程。

设(A,B)是一个交互协议，$(n,l,\alpha,\beta)\in SM$ 是它们的公共输入，$m=|n|$，$|n|$表示 n 的二进制表示的长度，B 检查是否 $n\geqslant 1, l\geqslant 1, \alpha,\beta\in Z_n^*, \alpha\neq\beta, \alpha^l=1$，如果不是，停机，否则，$A$ 和 B 重复执行下列步骤 m 次：

(1)A 随机选择一个整数 $j\in Z_l$，计算 $\gamma=\alpha^j \bmod n$，并将 γ 发送给 B；

(2)B 随机选择一个整数 $i=0$ 或 1，并将 i 发送给 A；

(3)A 计算 $h=(j+ik)\bmod n$，这里 $k=\log_\alpha^\beta$，并将 h 发送给 B；

(4)B 验证是否 $\alpha^h\equiv\beta^i\gamma \bmod n$。

如果在 m 轮中的每一轮都有 $\alpha^h\equiv\beta^i\gamma \bmod n$，那么接收 A 的证明，即认为$(n,l,\alpha,\beta)\in SM$。

前面我们给出的协议(A,B)都要求 A 和 B 之间依次执行若干轮，我们也可以要求 A 和 B 并行地一次把这些轮全部完成。前者通常称为串行协议，对应的零知识证明称为串行零知识证明。后者通常称为并行协议。对应的零知识证明称为并行零知识证明。二者之间的差别参见文献[23]。本小节讨论的证明系统中，只有一个证明者 A，也可以将此推广到多个证明者的情况，即证明者不只一个，详见文献[15]。关于成员的零知识证明就谈这些，感兴趣的读者可参见文献[5,12,13,14,15]。在这里值得一提的是，成员的零知识证明是针对某种语言而言的，究竟哪些语言具有成员的零知识证明还不太清楚。目前已证明[13]，如果单向置换存在，那么每一个 NP 问题都存在一个(计算)零知识证明系统。人们认为存在零知识证明系统的语言类要比 NP 类大，但没有证实这一点。例如图的非同构问题是否属于 NP 类，至今没有肯定的回答，但图的非同构问题存在一个(计算)零知识证明系统。

3.2.1.2 知识的零知识证明

我们知道，在成员的零知识证明中，证明者向验证者泄露了 1 比特的信息，即 $x\in L$。但在某些应用场合，要求证明系统不能泄露任何信息。例如，在大多数现有的识别技术(如 ID 卡、信用卡、计算机口令、PIN 号等)中，证明者 A 通过提交记在心中或印在卡上的一个词 $i(A)$来证明他的身份。这样，一个与不诚实的验证者合作的敌手就能得到该卡的一个拷贝或者知道了这个词 $i(A)$，从而敌手可使用 $i(A)$来假扮 A 享受 $i(A)$所允许的访问或服务。解决这个问题的一个办法是证明者 A 使用零知识证明来使验证者 B 相信他知道 $i(A)$，而不泄露 $i(A)$的任何一个比特。这种零知识证明就是本节我们将要介绍的知识的零知识证明。

设(A,B)是一个交互协议，A 和 B 均具有多项式时间的计算能力，公共输入为 $x\in\{0,1\}^*$，A 拥有一条私有知识带 S。设 P 是$\{0,1\}^*$上的一个多项式时间的二元关系(这个关系是公开知道的)。我们说 P 是$\{0,1\}^*$上的一个二元关系，是指 P 是$\{0,1\}^*\times\{0,1\}^*$的一个子集。如果$(x,w)\in P$，称 x 和 w 满足关系 P，记为 $P(x,w)=$真，否则，称 x 和 w 不满足关系 P，记为 $P(x,w)=$假。我们说 P 是$\{0,1\}^*$上的一个多项式时间的二元关系，是指 P 是满足下列两个条件的$\{0,1\}^*$上的一个二元关系：(1)$|w|$不超过$|x|$的多项式；

(2)对任何 $x,w\in\{0,1\}^*$，可在 $|x|$ 的多项式时间内检测出是否 $P(x,w)=$真。例如，我们可以考虑下列的关系 $P(x,w)$：w 是 x 模一个素数 Q 的离散对数或 w 是 x 的一个完全分解。

我们说 (A,B) 对多项式时间关系 P 是一个知识的交互证明系统(interactive proof system of knowledge)，如果它满足下列两个条件：

(1)完全性(completeness)：对所有的充分长的 x，如果在 A 的知识带 S 上存在一个 w 使得 $P(x,w)=$真，并且 A 和 B 都遵循协议，那么 B 将以很大的概率接收 x，即对每一个 $k>0$ 和充分长的 x：存在 $w\in S$，使 $P(x,w)=$真，B 至少以 $1-|x|^{-k}$ 的概率接收 x(即相信 A 知道使 $P(x,w)=$真的 w)。

(2)合理性(soundness)：对每一个 $k>0$，存在一个多项式时间的概率图林机 M(M 被允许在没有修改或检查它的带子的情况下，可重置和重新运行 A' 多项式次)，使得对每一个 $c>0$，对所有的 A'(可能不诚实)，A' 的随机带 RA' 和充分长的 x，如果将 x 作为 (A',B) 的公共输入，B 至少以 $|x|^{-k}$ 的概率接收 x，那么将 x 作为 (A',M) 的公共输入，(A',M) 至少以 $1-|x|^{-c}$ 的概率输出一个 w' 使得 $P(x,w')=$真。

条件(1)是说，如果 A 知道 w，B 将以很大的概率接收 A 对 x 的证明。条件(2)是说，如果 A 不知道 w，B 将以很小的概率接收 A 对 x 的证明。我们说 A 知道 w，是指存在某一多项式时间概率图林机 M(M 被允许在没有修改或检查它的带子的情况下，可重置和重新运行 A 多项式次)，使得 M 和 A 交互的结果是 w。

我们说一个协议 (A,B) 对多项式时间关系 P 是完全(统计、计算)零知识的，如果对任何多项式时间的概率图林机 B'(B' 带有一条附加输入带，B' 的附加输入记为 H)，随机变量族

$$\mathrm{View}_{A,B'}=\{\mathrm{View}_{A,B'}(x,H)\}_{(x,H)\in L'}$$

关于语言

$$L'=\{(x,H)\mid \text{存在 } w\in S,\text{使 } P(x,w)=\text{真},|H|\text{ 不超过 }|x|\text{ 的多项式}\}$$

是完全(统计、计算)可逼近的。我们说 (A,B) 对多项式时间关系 P 是一个知识的完全(统计、计算)零知识证明系统，如果它对 P 是一个知识的交互证明系统并且是完全(统计、计算)零知识的。

由定义可知，知识的完全零知识证明系统一定是知识的统计零知识证明系统，知识的统计零知识证明系统一定是知识的计算零知识证明系统。通常将知识的计算零知识证明系统称为知识的零知识证明系统，简称知识的零知识证明。

文献[16]中讨论了一种与 FFS 模型相类似的模型，在该模型中证明者 A 能证明 $I\in L$ 或 $I\notin L$，验证者 B 知道 $I\in L$ 还是 $I\notin L$，并相信 A 的证明，除此之外，B 没有从 A 那里获得任何附加信息。但被动的窃听者 C(不允许参加或插入协议)不能确定 $x\in L$ 还是 $x\notin L$，并且不能确定 A 的证明是否可信。关于知识的零知识证明的定义已给出了许多种[9,17]，本小节介绍的是 FFS 的定义[9]，文献[9]中也说明了知识的零知识证明系统的存在性，它指出：在 NP∩Co_NP 中的任何问题都有一个知识的零知识证明系统。不过由 FFS 的定义易知文献[13]中关于所有 NP 问题的成员的交互零知识证明也是知识的交互零知识证明。关于知识的零知识证明也可参考文献[18,19]。知识的零知识证明在密码协议的研究中有着重要的作用，关于密码协议我们将在后面的章节中要进行详细的介绍，作

为知识的零知识证明的应用，我们在这里暂举一例。

下面我们来描述一个身份识别方案，身份识别方案是一个协议，这个协议的目的是能使证明者 A 向验证者 B 证明他的身份，而事后 B 又不能冒充 A。这里假定存在一个可信任的中心，该中心的唯一目的是公开两个形式为 $4r+3$ 的大素数的乘积 n，这种整数称作 Blum 整数，在各种密码应用中通常使用这种整数作为模。Blum 整数的最有用的特性之一是：-1 是模 n 的一个非二次剩余，并且 -1 的 Jacobi 符号为 $+1$，即 $\left(\frac{-1}{n}\right)=1$。在公开 n 后，中心可以被取消或关闭，因为它再没有其它作用。方案中的每个人都能使用 n，但没有人能知道 n 的分解。

A 的秘密身份证 $i(A)$ 的产生过程如下：

(1)在 Z_n 中随机选择 k 个数 $S_1,S_2,\cdots,S_k$；

(2)随机地、独立地选择 $I_j=\pm 1/S_j^2 \bmod n, 1\leqslant j\leqslant k$；

(3)公开 $I_1,I_2,\cdots,I_k$，将 $P_i(A)=(I_1,I_2,\cdots,I_k)$ 作为 A 的公开身份证；保密 $S_1,S_2,\cdots,S_k$，将 $i(A)=(S_1,S_2,\cdots,S_k)$ 作为 A 的秘密身份证。

验证者 B 知道公开的 n 和 $P_i(A)$，证明者 A 想使 B 相信他知道 $i(A)$，但又不想对 B 泄漏任何信息。为了达到这一点，A 和 B 重复执行下列步骤 t 次(轮数 t 能降低 A 的欺骗概率)：

(1)A 选择一个随机数 R，计算 $X=\pm R^2 \bmod n$，并将它们之中的一个发送给 B；

(2)B 随机选择一个向量 $(E_1,\cdots,E_k)\in Z_2^k$，并发送给 A；

(3)A 计算 $Y=R\prod\limits_{\substack{E_j=1\\ 1\leqslant j\leqslant k}} S_j \pmod n$，并将 Y 发送给 B；

(4)B 验证是否 $X=\pm Y^2\prod\limits_{\substack{E_j=1\\ 1\leqslant j\leqslant n}} I_j \pmod n$

在上述的身份识别过程中，A 没有提交他的秘密身份证，也没有去向 B 证明他的公开身份证的合法性，而是向 B 通过显示他拥有关于他的秘密身份证的知识来证明他的公开身份证的合法性。

文献[9]中证明了在假定求模 n 的平方根困难的条件下，上述协议是 $\{S_j\}_{j=1}^k$ 的知识的零知识证明，其中 $k=O(\log(\log n))$，$t=O(\log n)$，n 的因子分解是未知的。

在前面的论述中，我们将协议中的参加者(如证明者、验证者、欺骗者等)都抽象成一台概率图林机。一台机器是由它所运行的程序或算法来控制的，程序或算法和机器之间是相互决定的，由此可见，可以将机器视作程序或算法，在现实世界中，有时也可以将机器视作人、现实的计算机等实体。

3.2.2 非交互零知识证明理论

现在我们考虑这样一种情况：A 和 B 是两个数学家，A 动身作一次长的环球旅行，在旅行期间他继续从事他的数学研究。我们希望 A 每发现一个新定理的证明都能通过给 B 写一张明信片用零知识证明他的论断的合法性。但是因为 A 没有固定或可预测的地址并且在任何邮件寄到 A 寄来的明信片上写的地址之前他已离开，所以即使 B 乐意与 A 交互，也无法与 A 取得联系。可见，前一节介绍的交互零知识证明已不再适合这种应用环

境，这里需要一个单项交互，即只是从 A 到 B。适用于这种应用环境的零知识证明就是下面我们将要介绍的非交互零知识证明(non-interactive zero-knowledge proof)。像交互零知识证明的研究一样，目前，在非交互零知识证明的研究中，人们也主要关心两种模型，一种是成员或定理的非交互零知识证明系统，另一种是知识的非交互零知识证明系统。本节我们只讨论成员或定理的非交互零知识证明系统，简称非交互零知识证明。对知识的非交互零知识证明系统，这里不作讨论，感兴趣的读者可参阅文献[20,21]。需指出的是知识的非交互零知识证明系统的构造不像知识的交互零知识证明系统的构造那样容易。

3.2.2.1 非交互证明系统

与交互证明系统相对应，在我们介绍非交互零知识证明之前首先来描述非交互证明系统。在一个非交互证明系统中也有两个分别称为证明者和验证者的参加者。证明者知道某一定理的证明，他希望向验证者证明他的确能证明这一定理。对一个语言 L 的非交互证明系统由两个阶段构成。第一个阶段是预处理阶段，主要建立证明者和验证者拥有的某些共同信息以及他们各自拥有的某些秘密信息，这个预处理阶段独立于定理证明阶段，而且允许证明者和验证者之间进行交互。第二个阶段是定理证明阶段，证明者选择并向验证者证明定理，这个定理证明阶段是非交互的。证明者和验证者可分别被考虑作 $P=(P_1,P_2)$ 和 $V=(V_1,V_2)$。P_1 和 P_2 是任意概率图林机(一般认为 P_2 具有无限的计算能力)，V_1 和 V_2 都是概率多项式时间图林机(也称多项式时间的概率图林机)。设 k 是一个安全参数。

我们说 $P=(P_1,P_2)$ 和 $V=(V_1,V_2)$ 对一个语言 $L\subset\{0,1\}^*$ 构成一个非交互证明系统，如果对所有充分大的 k，下列两个阶段的要求都满足：

(1)预处理阶段：通过 P_1 和 V_1 的交互产生三个输出 σ,S_P,S_V，其中 σ 是 P_1 和 V_1 共同拥有的公共信息(亦称参考串)，S_P 是 P_1 的秘密信息，S_V 是 V_1 的秘密信息。如果 P_1 不采纳规定的协议，那么 V_1 以很高的概率输出“拒绝”。

(2)定理证明阶段(所谓“定理”意指手边的一个串 $x\in L$)：P_2 非交互地向 V_2 证明定理(关于任何输入的通信仅是从 P_2 到 V_2 的一个消息)。

完全性：对每一个 $x\in L\cap\{0,1\}^k$，$P(V_2(1^k,\sigma,S_V,x,\beta)=\text{接收};(\sigma,S_P,S_V)\leftarrow(P_1\leftrightarrow V_1)_{(1^k)};\beta\leftarrow P_2(1^k,\sigma,S_P,x))\geqslant 1-2^{-2k}$，即 V_2 以很高的概率接收 P_2 的证明。

合理性：对每一个 $x\in\bar{L}\cap\{0,1\}^k$ 及每一个概率图林机 P_2'，$P(V_2(1^k,\sigma,S_V,x,\beta)=\text{接收};(\sigma,S_P,S_V)\leftarrow(P_1\leftrightarrow V_1)_{(1^k)};\beta\leftarrow P_2'(1^k,\sigma,S_P,x))\leqslant 2^{-2k}$，即 V_2 以很小的概率接收 P_2' 的证明。其中，$\bar{L}$ 表示 L 的补，即不属于 L 中的元素所组成的集合。1^k 表示 k 个 1 的级联。$(P_1\leftrightarrow V_1)_{(x)}$ 表示关于输入 x 的输出的概率空间，在这些输出中，也许一些是 P_1 或 V_1 的秘密信息，而另一些是它们的共同信息。如果 S 是一个概率空间，那么 $x\leftarrow S$ 表示根据 S 的概率分布随机选择一个元素分配给 x 的一个算法。如果 S 是一个有限集合，那么 $x\leftarrow S$ 表示从 S 中均匀地随机选择一个元素分配给 x 的一个算法。$P(p(x,y,\cdots):x\leftarrow S,y\leftarrow T,\cdots)$ 表示在依次执行完算法 $x\leftarrow S,y\leftarrow T$ 等后，关系 $p(x,y,\cdots)$ 是真的概率。

注意：(1)上述定义是一个更一般的定义，预处理阶段获得的 σ,S_P,S_V 中的某些可以是空串。(2)在某些情况下，也允许非交互证明系统把以前的定理及它们的证明的历史作为一个附加输入。

下面我们将对一类特殊语言 L 描述一个非交互证明系统，在此之前，先介绍一点预备知识。

关于合数的测试有如下结果[22]：存在一个多项式时间的概率算法 TEST(·,·)使得：(1)如果 x 是合数，那么在集合 $A=\{r|r$ 是一个串并且 r 的长度为 $|x|\}$ 中，至少有 3/8 使得 TEST(x,r)=合数。(2)如果 x 是素数，那么对所有的 r，都有 TEST(x,r)=素数。

整数 x 的长度是指 x 的二进制表示的串的长度，表示为 $|x|$。要特别注意在不同的场合，$|x|$表示的含义不同。如果 x 是一个实数，$|x|$表示 x 的绝对值。如果 x 是一个串，$|x|$表示 x 的长度。如果 x 是一个集合，$|x|$表示 x 中元素的个数。

对正整数 x，我们用 J_x^{+1} 和 J_x^{-1} 分别表示 Z_x^* 中 Jacobi 符号为+1 和−1 的集合。关于 Jacobi 符号及其性质参见附录。

我们说一个正整数 x 是正规的，如果 $|J_x^{+1}|=|J_x^{-1}|$。用 Regular(s)表示具有 s 个不同素因子的全体正规整数之集。关于正规整数，由 Jacobi 符号的定义易知下列结论成立：一个奇整数 $x\in$Regular(s)，当且仅当它有 s 个不同的素因子并且不是一个完全平方数。

设奇整数 $x\in$Regular(s)，定义等价关系 $\sim_x$ 为：$y_1\sim_x y_2$，当且仅当 y_1y_2 是模 x 的一个二次剩余。则 Z_x^* 由等价关系 $\sim_x$ 分成 2^t 个等价类且所有的等价类含有的元素个数都相同。特别地，J_x^{+1} 由等价关系 $\sim_x$ 分成 2^{t-1} 个等价类且所有的等价类含有的元素个数都相同，都等于模 x 的二次剩余的个数。对 $n\in N$，设

SQNR$(n)=\{(x,y)|x\in$Regular(2)，x 为奇整数，$|x|\leqslant n$，$y\in J_x^{+1}$ 但 y 不是模 x 的一个二次剩余$\}$，SQNR$=\bigcup_{n\geqslant 1}$SQNR(n)

这里 $|x|$表示整数 x 的长度。设 $(x,y)\in$SQNR，$z\in J_x^{+1}$，我们说 $s\in Z_x^*$ 是 z 的一个 (x,y) 根，如果 $z=s^2 \bmod x$ 或 $zy=s^2 \bmod x$，简记为 $s=\sqrt[(x,y)]{z}$。

下面就来描述一个对语言 SNQR 的非交互证明系统。

(1)预处理阶段：P_1 和 V_1 共同拥有一个 n^3 比特随机串 σ(即参考串)，设 $\sigma=\sigma_1\sigma_2\cdots\sigma_{n^3}$，每个 $\sigma_i(1\leqslant i\leqslant n^2)$的长度为 n，S_P 和 S_V 都是空串。

(2)定理证明阶段：P_2 非交互地向 V_2 证明定理。P_2 和 V_2 的共同输入为 $(x,y)\in$SQNR(n)，P_2 和 V_2 的执行规则分别如下：

P_2 的执行规则为：对 $i=1,2,\cdots,n^2$，如果 $\sigma_i\in J_x^{+1}$，那么随机地选择并发送 $s_i=\sqrt[(x,y)]{\sigma_i}$。

V_2 的执行规则为：

$V_2.1$ 验证 x 和 y，如果 x 是奇正整数并且 $y\in J_x^{+1}$，就继续执行 $V_2.2$，否则就停机并拒绝接收；

$V_2.2$ 验证 x 是不是一个完全平方数，如果不是，就继续执行 $V_2.3$，否则就停机并拒绝接收；

$V_2.3$ 验证 x 是不是一个素数幂，如果不是，就继续执行 $V_2.4$，否则就停机并拒绝接收；

$V_2.4$ 对每一个 $\sigma_i\in J_x^{+1}$，如果 $s_i=\sqrt[(x,y)]{\sigma_i}$，$V_2$ 就接收 P_2 的证明，否则就停机并拒绝接收。

因为 Jacobi 符号能在多项式时间内计算出，所以 V_2 的执行步骤 $V_2.1$ 和 $V_2.4$ 可在多

项式时间内验证。V_2 的执行步骤 $V_2.2$ 显然可在多项式时间内验证。而 $V_2.3$ 可由下列步骤来完成：

$V_2.3.1$ 计算整数 α 和 w 使得 $x=w^\alpha$。如果存在整数 α 和 w 使得 $x=w^\alpha$，那么 α 的取值范围只能是 $1,2,\cdots,|x|$，通过搜索能找到整数 α 和 w 使得 $x=w^\alpha$。

$V_2.3.2$ 如果对所有的 $1\leqslant i\leqslant n^2$，TEST$(w,\sigma_i)$=素数，那么 V_2 停机并拒绝接收。

由执行步骤 $V_2.3.1$ 和 $V_2.3.2$ 以及合数的测试算法 TEST$(\cdot,\cdot)$知 $V_2.3$ 可在多项式时间内完成。综上所述，V_2 是多项式时间的。

现在我们来证明上述系统满足完全性和合理性。

完全性：如果$(x,y)\in$SQNR(n)，那么 $V_2.1$ 和 $V_2.2$ 显然以概率 1 通过验证。下面着重讨论 $V_2.3$ 和 $V_2.4$ 通过验证的概率。

对任何固定的整数 $\bar{x}\in$Regular(2)，合数的测试算法 TEST$(\cdot,\cdot)$关于 σ_i 输出素数的概率至多是 $\frac{5}{8}$，这样 $V_2.3$ 不能通过验证的概率至多是 $(5/8)^{n^2}$。因为对某一个 w，至多有 2^n 个 x 使得$(x,w)\in$SQNR(n)，所以 $V_2.3$ 不能通过验证的概率至多是 $2^n\times(5/8)^{n^2}$。当 n 充分大时，$2^n\times(5/8)^{n^2}\leqslant 2^{-2n}$。故当 n 充分大时，$V_2.3$ 通过验证的概率不小于 $1-2^{-2n}$。

当 $x\in$Regular(2)时，由前面的讨论可知，在 J_x^{+1} 中有 2 个等价类，也就是 σ_i 或者是模 x 的一个二次剩余或者是与 y 在一个等价类中，在后者这种情况下，$y\sigma_i$ 是模 x 的一个二次剩余，因此，$V_2.4$ 以概率 1 通过验证。

由上述讨论知，对充分大的 n，当$(x,y)\in$SQNR(n)时，V_2 以不小于 $1-2^{-2n}$的概率接收 P_2 的证明，说明此系统满足完全性。

合理性：如果$(x,y)\notin$SQNR$(|x|\leqslant n,|y|\leqslant n)$，那么当 x 不是奇正整数或 $y\notin J_x^{+1}$ 时，显然，$V_2.1$ 通不过验证的概率为 1。当 x 是奇正整数并且 $y\in J_x^{+1}$ 时，有下述两种情况：(a)$x\in$Regular(2)，但 y 是模 x 的一个二次剩余；(b)$x\notin$Regular(2)。在情况(a)下，对任何固定的输入$(\bar{x},\bar{y})$，$V_2.4$ 能通过验证的充要条件是所有的 $\sigma_i(1\leqslant i\leqslant n^2)$都是模 $\bar{x}$ 的二次剩余，而 $J_{\bar{x}}$由等价关系 $\sim_{\bar{x}}$ 分成含相同个数的 $2^2=4$ 个等价类，所以每个 σ_i 是模 $\bar{x}$ 的一个二次剩余的概率至多是 1/4，这样 $V_2.4$ 能通过验证的概率至多是$\left(\frac{1}{4}\right)^{n^2}=2^{-2n^2}$。因为对某一固定的 $\bar{y}$，至多有 2^n 个 $\bar{x}(|\bar{x}|\leqslant n)$使得$(\bar{x},\bar{y})\notin$ SQNR 且 $\bar{y}$ 是模 $\bar{x}$ 的一个二次剩余，所以 $V_2.4$ 能通过验证的至多是 $2^n\times 2^{-2n^2}=2^{-2n^2+n}$。当 n 充分大时，$2^{-2n^2+n}\leqslant 2^{-2n}$，因此，在情况(a)下，$V_2.4$ 能通过验证的概率不大于 2^{-2n}。在情况(b)下，或者(b.1)x 不是正规整数，或者(b.2)，$x\in$Rugular(1)，或者(b.3)$x\in$Regular(s)，$s\geqslant 3$。在情况(b.1)下，因为一个奇正整数 x 一定是一个完全平方数，这将在 $V_2.2$ 中被检测出。在情况(b.2)下，x 是一个素数幂，将在 $V_2.3$ 中被检测出。在情况(b.3)下，对任何固定的$(\bar{x},\bar{y})$，$\bar{x}\in$Regular(s)，$s\geqslant 3$，$V_2.4$ 能通过验证的充要条件是对每一个 $\sigma_i\in J_x^{+1}$，或者 σ_i 或者 $\sigma_i\bar{y}$ 是模 $\bar{x}$ 的一个二次剩余，而 J_x^{+1} 至少有 4 个等价类，因而 σ_i 或者 $\sigma_i\bar{y}$ 是模 $\bar{x}$ 的一个二次剩余的概率至多是 1/2。这样，$V_2.4$ 能通过验证的概率至多是$(1/2)^{n^2}=2^{-n^2}$。因为至多有 2^{2n}对$(x,y)\notin$SQNR$(|x|\leqslant n,|y|\leqslant n)$，所以对任何输入 $V_2.4$ 能通过验证的概率至多是 $2^{2n}\times 2^{-n^2}=2^{-n^2+2n}$，当 n 充分大时，$2^{-n^2+2n}\leqslant 2^{-2n}$。故在情况(b)下，$V_2.4$ 能通过验证的概率不大于 2^{-2n}。综上所述，此

系统满足合理性。

3.2.2.2 非交互零知识证明

非交互零知识证明是一种特殊的非交互证明系统，它要求在证明中不允许泄露任何有用的消息。像交互零知识证明系统一样，非交互零知识证明系统的证明者能使验证者相信 x 具有某种具体的特性(诸如 $x\in L$)，但执行完协议后，验证者自己仍然一点也不知道如何来证明 x 具有这个特性。

为了定义零知识性，先介绍一些记号。

对任何图林机 V_1'，设 $(P_1,V_1')(1^k)$ 表示由 P_1 和 V_1' 交互的输出、交互的历史和 V_1' 的掷硬币所构成的概率空间，把交互的历史和 V_1' 的掷硬币合成一个元素 h。用 (σ,S_P,S_V,h) 表示这个空间的一个元素。

设 $P=(P_1,P_2)$，$V=(V_1,V_2)$ 是关于语言 L 的一个非交互证明系统。对每一对概率多项式时间图林机 $V'=(V_1',V_2')$(允许它们有附加的输入带)和 $x\in L\cap\{0,1\}^k$，将 V' 对 P 关于输入 x 的观察记为 $\mathrm{View}_{V'}^{(P)}(1^k,x)$，这里

$$\mathrm{View}_{V'}^{(P)}(1^k,x)=\{(\sigma,S_v,h,\mathrm{proof}):(\sigma,S_P,S_V,h)\leftarrow(P_1,V_1')(1^k);$$
$$\mathrm{proof}\leftarrow P_2(1^k,\sigma,S_P,x)\}$$

是一个随机变量。

我们说一个关于语言 L 的非交互证明系统 $P=(P_1,P_2)$，$V=(V_1,V_2)$ 是完全(统计、计算)零知识的，如果对每一对概率多项式时间图林机 $V'=(V_1',V_2')$，存在一个概率多项式时间图林机 S 使得对每一个充分大的 k 及每一个 $x\in L\cap\{0,1\}^k$，$\mathrm{View}_{V'}^{(P)}(1^k,x)$ 和 $S(1^k,x)$ 是完全(统计、计算)不可区分的。

注意：(1)允许非交互零知识证明系统把以前的定理及其证明历史、交互历史作为一个附加输入，这里为了简单起见，没有考虑。

(2)由定义易知，非交互完全零知识证明系统一定是非交互统计零知识证明系统，非交互统计零知识证明系统一定是非交互计算零知识证明系统。

(3)如果证明者多次使用同一个参考串 σ 使验证者相信许多定理的正确性，那么产生的非交互证明系统也许不再是零知识的，这种性质称为有界性，具有这种性质的非交互零知识证明系统称为有界的非交互零知识证明系统，3.2.2 节 1 小节中的例子就是一个有界的非交互完全零知识证明系统，下面我们将证明其零知识性。

为了说明 3.2.2 节 1 小节中的非交互证明系统是(完全)零知识的，首先我们构造一个概率多项式时间图林机(模拟器)S，其运行程序如下：

输入 $(x,y)\in\mathrm{SQNR}(n)$。

$S.1$ 设 proof＝空串。

$S.2$ 对 $i=1,2,\cdots,n^2$，随机地选择一个 n 比特整数 s_i，如果 $s_i\notin J_x^{+1}$，那么设 $\sigma_i=s_i$，否则掷一个硬币，如果是 0，设 $\sigma_i=s_i^2\bmod x$ 并把 s_i 级联到 proof，如果是 1，设 $\sigma_i=y^{-1}s_i^2\bmod x$ 并把 s_i 级联到 proof；

$S.3$ 设 $\sigma=\sigma_1\sigma_2\cdots\sigma_{n^2}$。

输出 (σ,proof)。

因为在这个非交互证明系统中，S_P 和 S_V 都是空串，所以欲证明这个系统是(完全)零

知识的，只需证明对所有的充分大的 n 和所有的 $(x,y)\in \mathrm{SQNR}(n)$，有

$$\mathrm{View}_W^{(P)}(1^n,(x,y)) = S(1^n,(x,y))$$

其中

$$\mathrm{View}_V^{(P)}(1^n,(x,y)) = \{(\sigma,\mathrm{proof}):\sigma\leftarrow\{0.1\}^{n^3};\mathrm{proof}\leftarrow P_2(1^n,\sigma,(x,y)\}$$

亦即证明由 S 输出的随机变量和由 V_2 看到的随机变量是完全一样的，这样，这两个随机变量不能被任何观察者(observer)或区分者(distinguisher)区别开来。

事实上，易知 σ 随机地分布在所有的 n^3-比特长的串上。另外，如果 $\sigma_i\in J_x^{+1}$，那么对应的 s_i 是 σ_i 的一个随机的 (x,y) 根。这样，关于输入 (x,y) 和 σ,s_i 属于 S 的输出的概率和属于从证明者 P 发送给验证者 V 的证明的概率是一样的。故

$$\mathrm{View}_V^{(P)}(1^n,(x,y)) = S(1^n,(x,y))$$

目前已有不少文献讨论了非交互零知识证明系统，主要讨论其实现问题，这在文献[8]中作了详尽的论述。文献[24]中解决了文献[7]中的两个公开问题，文献[25]中也独立地解决了文献[7]的第一个公开问题。文献[24]证明了任何单向置换能代替二次剩余并证明了对实现有界的非交互零知识证明单向置换是充分的，但证明者需要指数计算能力。文献[26]指出了文献[24]的漏洞并说明了如何通过一个附加的证书来修改这个漏洞。文献[27]说明了非交互零知识证明和不变数字签名的等价性。如果任何单向函数存在，文献[28]指出，在一个交互的预处理阶段后，任何充分短的定理能被非交互地、零知识地证明，而文献[29]指出，在通过执行一个健忘传输协议的预处理阶段后，任何定理能被非交互地、零知识地证明。

3.2.2.3 公开可验证的非交互零知识证明

由证明者 P 所提供的一个定理的证明，关于它的验证有两种可能性：(1) 定理的证明被直接提供给一个特定的验证者 V 而且只有 V 才能验证；(2) 定理的证明能被系统中的任何用户验证。在后一种情况下，我们称证明是公开可验证的。

我们说一个非交互零知识证明系统是公开可验证的，如果这个系统的证明者所提供的证明是公开可验证的。

公开可验证的非交互零知识证明系统的重要性在于它可以应用于数字签名和消息认证等密码协议之中，证明的公开可验证性将对应于任何人能检验签名。文献[30,31]提出了实现公开可验证的非交互零知识证明系统的一些方法，这些系统都允许在非交互地、零知识地证明定理之前，许多证明者和验证者无需相互交互，但文献[6,7,8]的系统限制证明者和验证者共享一个参考串来非交互地进行零知识证明。这样，如果许多用户希望互相证明定理，那么它们中的每对都不得不共享一个参考串，参考串的数量随用户的数量的增加而迅猛地增加，这是不切实际的。

在需要执行大量密码协议的大型网络中，非交互零知识证明是十分有用的，这是因为在这些协议中零知识是十分关键的，同时交互需要花更大的代价，一个典型密码协议的内部计算能在几秒钟内完成，而交换电子邮件所花的时间是它的上百倍。这样，在密码协议中，非交互零知识证明也许被用来节省昂贵的通信圈。文献[32]讨论了非交互零知识证明在公钥密码系统中的应用，文献[26]对此也作了进一步讨论。文献[30]介绍了非交互零知识证明在数字签名、消息认证和识别号的无记忆分配等方面的一些应用。

3.3 注记和文献

算法与问题的复杂性理论是现代密码体制设计与分析的基础。限于篇幅，本章仅在非形式的水平上进行了讨论，有兴趣的读者可参阅文献[2,3]。目前已经确认的 NP 完全问题有 300 多个，大部分都可以在文献[3]中找到。关于算法与问题复杂性理论与密码之间的关系将在以后的章节中逐步介绍。另外，文献[35,36]也对算法与问题的复杂性理论作了一些简单介绍。

零知识证明理论在密码体制的设计与分析中占有重要的地位，特别是在密码协议的设计与分析中尤为重要。本章用大量篇幅较系统地介绍了零知识证明理论的一些基本知识。关于零知识证明理论的一些发展和应用在各章节已作了一些介绍，感兴趣的读者可在计算机科学基础研讨会(Foundation of Computer Science (FOCS) Conference)、计算理论专题研讨会(Symposium on the theory of Computing (STOC) conference)、各种密码会议(如 Crypto conference, Eurocrypt conference 等)等的论文集中找到很多有关这个主题的论文。

关于零知识证明理论的综述性文章可参阅文献[34,37,38]。

参 考 文 献

[1] Cook, S. A., The Complexity of Theorem-proving Pro-cedures, Por. 3rd Annual ACM Symp. On the Theory of Computing, pp. 151—158, 1971.

[2] Aho, A., Hopcraft, J. and Ullman, J., The Design and Analysis of Computer Algorithms, Addison-Wesley, 1974.

[3] Garey, M. R. and Johnson, D. S., Computers and Intractablity: A Guide to the Theory of NP-Completeness, W. H. Freeman and Co., 1979.

[4] Goldwasser, S., Micali, S. And Rackoff, C., The Knowledge Complexity of Interactive Proof Systems, Proceedings of the 17th ACM Symposium on the Theory of Computing, 1985. pp. 291—304.

[5] Goldwasser, S., Micali, S. And Rackoff, C., The Knowledge Complexity of Interactive Proof Systems, SIAM J. Comput, 18(1989), pp. 186—208.

[6] Blum, M., Feldman, P. and Micali, S., Non-interactive Zero-knowledge Proof Systems and Applications, In Proc. 20th Annual ACM Symposium on Theory of Computing, Chincago, IL, 1988, pp. 103—112.

[7] De Santis, A., Micali, S. and Persiano, G., Non-interactive Zero-knowledge Proof Systems, Advances in Cryptology-Crypto'87, Springer-Verleg, Berlin, New York, 1987, pp. 52—72.

[8] Blum, M., De Santis, A., Micali, S. And Persiano, G., Non-interactive Zero- knowledge, SIAM J. Comput., 20 (1991), pp. 1084—1118.

[9] Feige, U., Fioot, A. And Shamir, A., Zero Knowledge Proofs of Identity, Proceedings of STOC, 1987, pp. 210-217. (J. Of Cryptology, Vol. 1, 1988, pp. 77—94)

[10] Quisquater, J.-J., Guillou, L. And Berson, T., How to Explain Zero-knowledge Protocols to Your Children, Advances in Cryptology-Crypto'89, Springer-Verleg, 1990, pp. 628—631.

[11] Balcazar, J., Diaz, J. And Gabarro, J., Structural Complexity I, II, EATCS Monographs on Theoretical Computer Science, Springer-Verlay, Berlin, 1988, 1991.

[12] Brassard, G., Chaum, D. And Crépeau, C., Minimum Disclosure Proofs of Knowledge, Journal of Computer and Systems Science, 37(1988), pp. 156—189.

[13] Goldreich, O., Micali, A. And Wigdeson, A., Proofs That Yield but Their Validit or All Languages in NP Have

Zero-knowledge Proof Systems, Journal of the ACM, 38(1991),pp. 691—729.

[14] Stinson, D. R., Crytography-Theory and Practice,CRC Press,1995.

[15] Ben-or, M., Goldwasser, S., Kilian, J., Wigderson, A., Multi-Prover Interactive Proofs: How to Remove Intractability Assumptions STOC 1988, pp. 113—131.

[16] Galil, Z., Haber, S. And Yung, M., A Private Interactive Test of a Boolean Predicate and Minimum-knowledge Public Key Cryptosystems, Proceeding of FOCS 1985,pp. 360—371.

[17] Bellare, M. And Goldreich, O., On Defining Proofs of Knowledge, Advances in Cryptology-Crypto'92, Springer-Verlag,Berlin,1993, pp. 390—420.

[18] Feige, U. And Shamir, A., Zero-knowledge Proof of Knowledge in Two Rounds, Advances in Cryptology-Crypto'89, Springer-Verlay,Berlin,pp. 526—544.

[19] Tompa, M. And Woll, H., Random Self-reducibility and Zero-knowledge Interactive Proofs of Possession of Information, Proceedings of the 28th Symposium on Foundations of Computer Science,1987,pp. 472—482.

[20] De Santis, A. And Persiano, G., The Power of Preprocessing in Zero-knowledge Proofs of Knowledge, J. Cryptology, Vol. 9, No. 3, pp. 129—148,1996.

[21] De Santis,A. And Persiano, G., Zero-knowledge Proofs of Knowledge Without Interaction, Proceeding of the 33rd Symposium on Foundations of Computer Science,1992,pp. 427—437.

[22] Rabin, M., Probabilistic Algorithm for Testing Primality, J. Number Theory, 12 (1980),pp. 128—138.

[23] Sakurai, K. And Itoh, T., On the Discrepancy between Serial and Parallel of Zero-knowledge Protocols, Advances in Cryptology-Crypto'92, Springer-Verlay,1993,pp. 264—259.

[24] Feige,U. Lapidot, A. And Shamir, A., Multiple Non-Interactive Zero-knowledge Proofs Based on a Single Random String, In Proc. 31st Annual IEEE Symposium on Foundations of Computer Science, St. Louis, Mo,1990, pp. 308—317.

[25] De Santis, A. And Yung, M., Cryptographic Applications of the Non-interactive Metaproof and Many-prover Systems, Advances in Cryptology-Crypto'90, Springer-Verlay,Berlin, NewYork, 1990, pp. 366—377.

[26] Bellare, M. And Yung, M., Certifying Cryptographic Tools: The Case of Trapdoor Permutations, Advances in Cryptology-Crypto'29, Springer-Verlay,Berlin, New York, 1992, pp. 442—460.

[27] Goldwasser, S. and Ostrovsky, R., Invariant Signatures and Non-interactive Zero-knowledge Proofs are Equivalent, Advances in Cryptology-Crypto'92, Springer-Verlay,Berlin, New York, 1992, pp. 228—245.

[28] De Santis, A., Micali, S. and Persiano, G., Non-interactive Zero-knowledge Proof-systems with Preprocessing, Advances in Cryptology-Crypto'88, Springer-Verlay,Berlin, New York, 1988, pp. 269—283.

[29] Kilian, J., Micali, S. and Ostrovsky, R., Minimum Resource Zero-knowledge Proofs, Advances in Cryptology-Crypto'89, Springer-Verlay,Berlin,New York, 1989,pp. 545—546.

[30] Bellare, M. and Goldwasser, S., New Paradigms for Digital Signatures and Message Authentication Based on Non-interactive Zero-knowledge Proofs, Advances in Cryptology-Crypto'89, Springer-Verlay, Berlin, New York, 1989, pp. 194—211.

[31] Bellare, M. And Micali, S., Non-interactive Obvious Transfer and Applications, Advances in Cryptology-Crypto'89, Springer-Verlay, Berlin, New York, 1989, pp. 547—559.

[32] Naor, M. And Yung, M., Public-key Cryptosystems Provable Secure against Choosen Ciphertext Attack, In Proc. 22nd Symposium on Theory of Computing, Baltimore, MD, 1990, pp. 427—437.

[33] Burmester, M. And Desmedt, Broadcast Interactive Proofs, Advances in Cryptology-Eurocrypt'91, Springer-Verlay, Berlin, New York ,1991,pp. 81—95.

[34] Simmons, G. J. (Edited by Simmons, G. J.), Contemporary Cryptology, IEEE Press,1991.

[35] 王育民、何大可,保密学-基础与应用,西安电子科技大学出版社,西安,1990.

[36] Denning, D. E. R., Cryptography and Data Security, Addiso-Wesley, 1982.

[37] 朱洪、吴京,零知识证明浅介,密码与信息,1992,No. 4,1—38 页.

[38] 冯登国,非交互零知识证明及其密码应用,密码与信息,1996,No. 2, 14—22 页.

第 4 章　私钥密码算法——流密码

在第 1 章中我们已经指出，私钥密码体制根据对明文消息加密方式的不同可分为两大类，即流密码和分组密码。在分组密码中，明文被分成 m 个符号的大数组 $x=(x_1,x_2,\cdots,x_m)$。每一组明文在密钥 $k=(k_1,k_2,\cdots,k_l)$的控制下变换成 n 个符号的密文组 $y=(y_1,y_2,\cdots,y_n)$，可简记为 $y=E_k(x)$，而且每组明文用同一个密钥 k 加密。诸如单表代换密码就是一种分组密码。一个流密码将消息分成连续的符号 $x=x_1,x_2,\cdots$，用密钥流 $k=k_1,k_2,\cdots$ 的第 i 个元素 k_i 对 x_i 加密，即 $E_k(x)=E_{k_1}(x_1)E_{k_2}(x_2)\cdots$。诸如 Vigenère 密码就是一种流密码。如果密钥流经过 d 个符号之后重复，则称该流密码是周期的，否则称之为非周期的。一次一密密码是非周期的；Vigenère 密码是周期的。

分组密码与流密码之间的主要区别在于记忆性（见图 4.0.1 所示）。

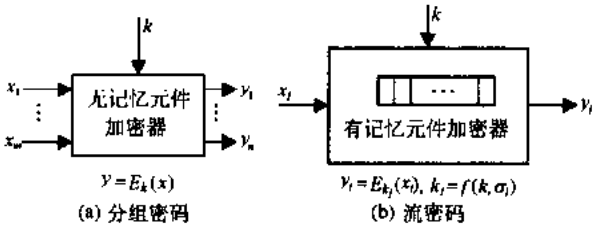


图 4.0.1　分组密码与流密码的加密方式

在流密码中，密钥流元素 k_j 的产生由第 j 时刻流密码的内部状态 σ_j 和被称作种子密钥（或实际密钥）的 k 所决定，一般可写为 $k_j=f(k,\sigma_j)$。加（或解）密变换 E_{k_j}（或 D_{k_j}）是时变的，其时变性由加（或解）密器中的记忆元件来保证。而分组密码的加（或解）密变换 E_k（或 D_k）不是时变的，加密器中不存在记忆元件。

本章主要来介绍有关流密码体制的一些研究工作，至于分组密码体制将在下一章介绍。

4.1　流密码的分类及其工作模式

在流密码中，加密器中存储器的状态随时间而变化，这一变化过程可用一个函数（通常称为状态转移函数）来描述，记为 f_s。根据状态转移函数 f_s 是否依赖于输入的明文符号（字符或比特），可将流密码分为两类，即同步流密码（synchronous steam cipher）和自同步流密码（self-synchronous stream cipher）。

在同步流密码中，状态转移函数 f_s 与输入的明文符号无关。此时，密钥流 $\{k_j=f(k,\sigma_j)\}_{j=1}^{\infty}$ 与明文符号无关，而 j 时刻输出的密文 $c_j=E_{k_j}(m_j)$ 也不依赖于 j 时刻之前的明文符号。因而可将同步流密码的加密器划分成密钥流生成器（或滚动密钥生成器（running key generator），或伪随机序列生成器（pseudorandom sequence generator））和加密变换器（单纯利用滚动密钥 k_j 对输入的明文符号 x_j 进行加密的变换器）两部分（见图 4.1.1）。

在同步流密码中，只要发送端和接收端有相同的（种子或实际）密钥 k 和内部状态，就能产生出相同的密钥流。此时，我们说发送端和接收端的密钥生成器是同步的。一旦二者不同步，解密工作立即失败，密码系统在这时要能提供某种辅助手段以重建同步。

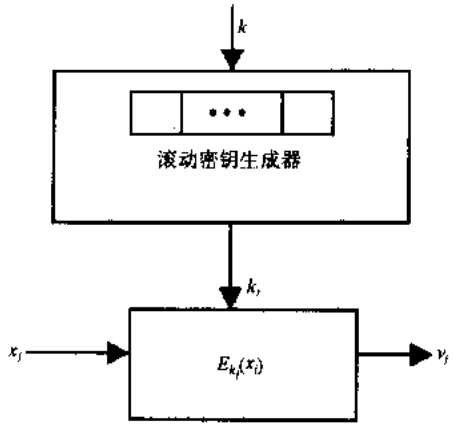


图 4.1.1　分解后的同步流密码

同步流密码有两种基本的工作模式。一种是输出-分组反馈模式[2](OFM)(见图 4.1.2)。反馈寄存器 R 作为密钥 B 所决定的分组加密算法的输入。在第 i 次迭代中,先计算 $E_B(R)$,然后将输出组的最低位(最右边)符号作为第 i 个密钥符号 k_i 输出。同时,整个输出组反馈到 R 中,作为下次迭代的输入。因为在整个密钥流的生成中反馈是在内部进行的,这种方式也称为内部反馈。在这种同步流密码中,分组加密算法 E_B 应选择为非线性变换,Gait[3]曾建议用 DES(见 5.2 节)作为这种流密码的非线性变换。

另一种是 Diffie 和 Hellman[4]给出的计数模式(见图 4.1.3)。在这种同步流密码中,分组加密算法 E_B 的输入由一个计数器提供。这种方法的优点在于,不用生成前面 $i-1$ 个密钥符号即可直接生成第 i 个密钥符号 k_i,其方法是直接将计数器置到 I_0+i-1。

同步流密码的一个优点是无错误传播,一个传输错误只影响一个符号,不会影响后继

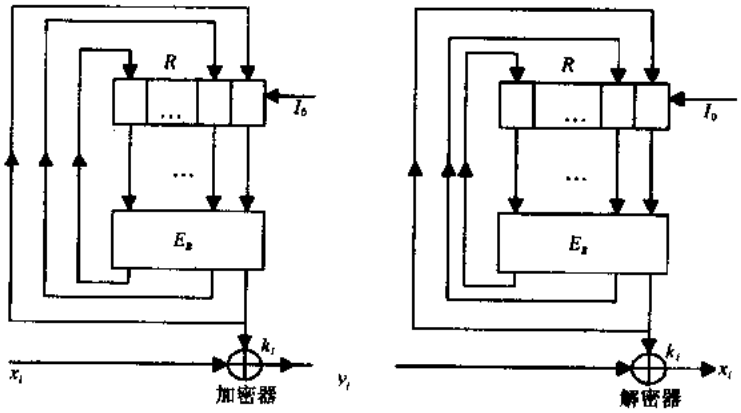


图 4.1.2　输出一分组反馈模式

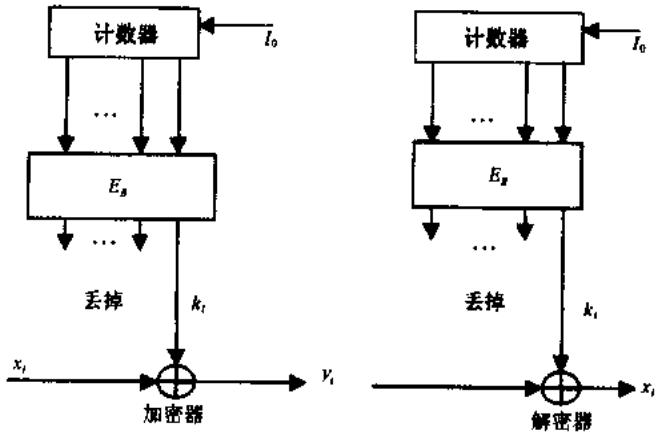


图 4.1.3　计数模式

符号。但这也是一个缺点，因为敌手窜改一个符号比窜改一组符号容易。通过附加非线性检错码可克服这个缺陷。

在自同步流密码中，状态转移函数 f_i 与输入的明文符号有关。此时，密钥流 $k_j=f(k,\sigma_j)$ 与明文符号有关，而 j 时刻的密文 c_j 不仅仅依赖于明文符号 x_j。这种思想可追溯到 16 世纪的 Vigenère 所发明的自身密钥密码[6]。Vigenère 自身密钥密码的密钥流通过在初始密钥 k_1 后附加每个密文符号来产生，即 $k_i=y_{i-1}(i>1)$。显然，这种密码是很不安全的。但用所加密的消息来产生非重复密钥流的思想对密码学却是一大贡献。

Vigenère 自身密钥密码的缺陷在于将密钥暴露在密文之中。将密文符号送到一个非线性分组密码来导出密钥符号可克服上述缺陷。这种技术称之为密码反馈模式(CFB)，因为密文符号参与了反馈圈。由于每个密文符号实际上依赖于前面的所有密文符号，这种技术有时也称作"链接"。这种模式已经由美国国家标准局批准用于 DES(见 5.4 节)。

密码反馈模式是自同步流密码的一种最常用的工作模式(见图 4.1.4)。每个密文符号 y_i 在生成之后，立即送到移位寄存器 R 的一端(另一端的符号被丢掉)。在每次迭代中，R 的值作为分组加密算法 E_B 的输入，而输出组的最低位符号用作下一个密文符号。

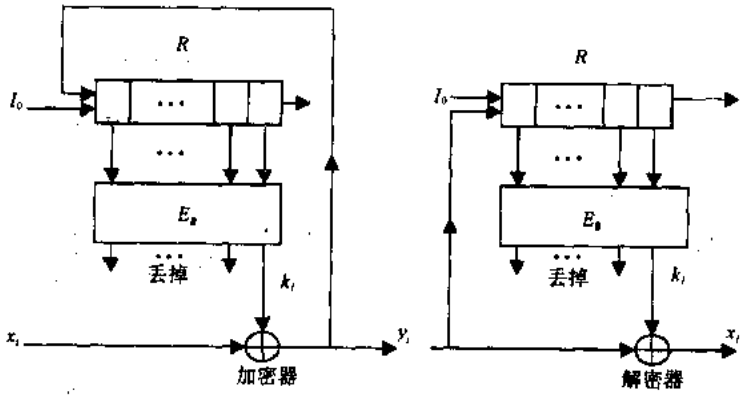


图 4.1.4 密码反馈模式(CFB)

对于密码反馈模式，传输错误影响反馈圈。如果一个密文符号在传输中出错或丢失，等到该错误移出寄存器才能同步。因而，一个错误至多影响 n 个符号，这里 n 为每组的符号个数。

在 2.1 节中所介绍的"一次一密"密码是当今流密码体制的原型。"一次一密"要求用户在安全信道中(预先)传送长度不小于明文消息符号数量的密钥符号，这样做不切实际。上面介绍的流密码体制都不是完善保密系统，但是它们应当是条件安全的或计算上不可破译的。为此，由(实际或种子)密钥 k 扩展成的密钥流序列 $\{k_j\}_{j=1}^{\infty}$ 应当满足一定的要求，诸如，极大的周期、良好的统计特性、能对抗已知的若干种攻击方法(如线性逼近、分别征服攻击等)等。

从公开发表的文献来看，目前的绝大多数有关流密码的研究成果都是同步流密码方面的。由于自同步流密码系统一般需要密文反馈，因而使得分析工作复杂化。但自同步流密码具有抵抗密文搜索攻击和认证功能等优点，所以这种流密码也是一个值得进一步研究的课题。

一个同步流密码是否具有很高的密码强度主要取决于密钥流生成器的设计。为了设

计安全的密钥流生成器，必须在生成器中使用非线性变换，这就给生成器的理论分析工作带来了很大困难。在图 4.1.1 的生成器中，状态转移函数和输出函数一般应为非线性变换。为了对某些这类生成器便于从理论上进行分析，Rueppel[1]将这类生成器分成两部分，即驱动部分和非线性组合部分。驱动部分控制生成器的状态序列，并为非线性组合部分提供统计性能好的序列。例如，驱动部分可由一组最大长度线性反馈移位寄存器组成。而非线性组合部分将驱动部分所提供的序列组合成密码学特性好的序列，例如用于提高密钥流的线性复杂度等(见图 4.1.5)。

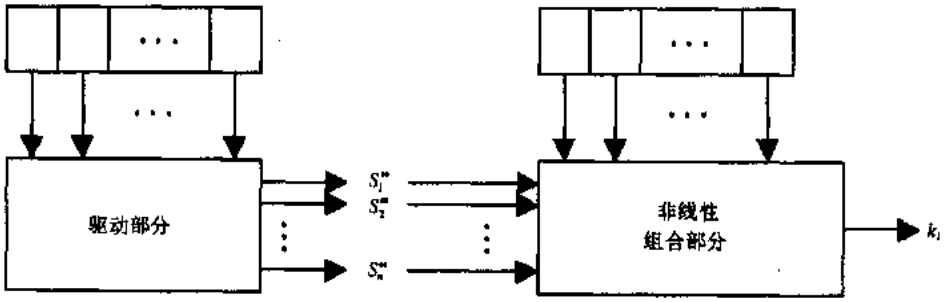


图 4.1.5　密钥流生成器的分解

用图 4.1.5 所示的密钥流生成器获取密钥流的做法与 Shannon 早期提出的两条密码原则——“扩散”和“混淆”恰好是一致的。在这里，驱动部分中的线性反馈移位寄存器(LFSR)将实际密钥 k 扩散(扩展)成周期很大的驱动序列，驱动序列与密钥 k 之间过分明显的依赖关系再经组合部分的适当非线性变换加以隐蔽，以实现所谓的“混淆”。

4.2　线性反馈移位寄存器和 B-M 算法

线性反馈移位寄存器(LFSR)因其实现简单、速率高、便于分析等优点而被广泛地用于各类数字电路中，也成为构造密钥流生成器的最重要的部件之一。本节我们主要来介绍有关二元域 $F_2=Z_2$ 上的线性反馈移位寄存器的一些基本结果，当然这些结果对一般的有限域上的线性反馈移位寄存器也成立，感兴趣的读者请参阅文献[5,6]。关于剩余类环 Z_N 上的线性反馈移位寄存器的研究参见文献[149,150]。

F_2 上 n 级 LFSR 的一般结构如图 4.2.1 所示。它是由 n 个二元存储器与若干个 F_2 上

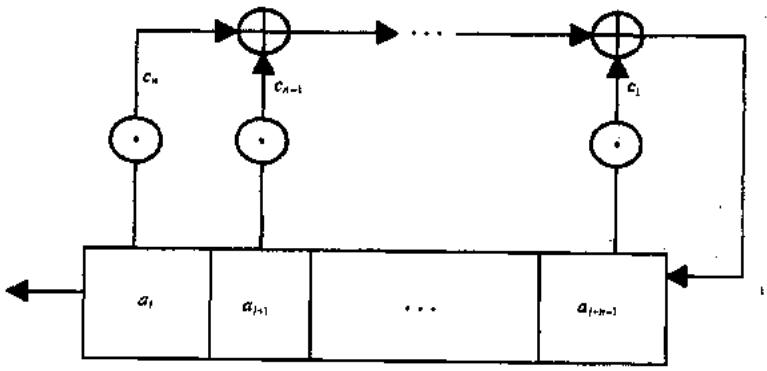


图 4.2.1　F_2 上的一个 n 级 LFSR

的乘法器和加法器连接而成(二元情况乘法器可以省略)。图中的 c_i, a_j 等都是 F_2 中的元素。

每一存储器称为 LFSR 的一级,n 为 LFSR 的级数或长度。在第 j 个移位时钟脉冲到来时,LFSR 的状态由 $(a_j, a_{j+1}, \cdots, a_{j+n-1})$ 变为 $(a_{j+1}, a_{j+2}, \cdots, a_{j+n})$,并送出 a_j 作为输出序列的一位。初始状态 $(a_0, a_1, \cdots, a_{n-1})$ 可由用户指定,而补入末端存储器的 a_{j+n} 之值由反馈函数或(n 阶)线性递归关系(如果将各个 a_k 视为未知量,也称为 n 阶线性齐次差分方程)

$$a_{j+n} = \sum_{i=1}^{n} c_i a_{j+n-i}, j \geqslant 0 \tag{4.2.1}$$

确定。用延迟算子 $D(Da_k = a_{k-1})$ 作为未定元给出的反馈多项式是

$$f(D) = 1 + c_1 D + c_2 D^2 + \cdots + c_n D^n$$

易知

$$f(D)a_k = 0 \qquad (k \geqslant n) \tag{4.2.2}$$

将此事实简记为 $f(D)a = 0$,约定用 a 表示输出序列 $a_0, a_1, a_2, \cdots$,通常称 a 为一个 n 阶线性递归序列(也称 n 级 LFSR 序列)。常用 x 表示未定元,称

$$f(x) = 1 + c_1 x + c_2 x^2 + \cdots + c_n x^n$$

为 LFSR 的联结多项式(connection polynomial)。$f(x)$ 的互反多项式

$$\overline{f(x)} = x^n f\left(\frac{1}{x}\right) = x^n + c_1 x^{n-1} + \cdots c_{n-1} x + c_n$$

称为 LFSR 的特征多项式(characteristic polynomial)。称描述序列 a 的形式幂级数

$$a(D) = a_0 + a_1 D + a_2 D^2 + \cdots = \sum_{j=0}^{\infty} a_j D^j$$

为 a 的生成函数或 D-变换。实际上,按自然推广的多项式加法与乘法,F_2 上的形式幂级数构成一个交换环,记为 $F_2\langle D\rangle$。多项式环 $F_2[D]$ 为其子环。于是可定义

$$P(D) = \sum_{i=0}^{\infty} p_i D^i = f(D) \cdot a(D) \tag{4.2.3}$$

但由式(4.2.2)易知,$p_i = 0, i \geqslant n$,即 $P(D)$ 是 F_2 上次数小于 n 的多项式。借助多项式的 Euclidean 算法即长除法,我们记

$$a(D) = \frac{P(D)}{f(D)} \qquad \partial P < n \tag{4.2.4}$$

式(4.2.4)称为线性递归序列 a 的基本恒等式,也称为 a 的有理分式表示。如果 $P(D)$ 与 $f(D)$ 互素,则称有理分式(4.2.4)为 a 的不可约有理分式表示。由于 $P(D)$ 的取法与 n 级 LFSR 初态的取法均为 2^n 种,因此,利用式(4.2.3)可建立二者之间的一一对应关系。这样,可以将式(4.2.4)看作差分方程(4.2.1)的通解。欲求反馈多项式为 $f(D)$ 的 LFSR 在指定初态 $(a_0, a_1, \cdots, a_{n-1})$ 下的输出序列 a,只需由式(4.2.3)定出 $P(D)$ 即可。这表明,LFSR 电路运行的机制或规律可以纯代数地加以研究。

定义 4.2.1 对于 F_2 上的半无限序列 $a = a_0 a_1 a_2 \cdots$,如果存在正整数 T 和非负整数 j_0,使得 $a_{j+T} = a_j$ 对所有 $j \geqslant j_0$ 都成立,则称该序列是终归周期的,且称 T 是该序列的一个周期。所有可能周期的最小者称为序列的最小周期,记为 $p(a)$。特别地,如果 $j_0 = 0$,则称该序列为周期序列(period sequence)。若 $p(a)$ 是 a 的最小终归周期,则称使 $a_{j+p(a)} = a_j$ 对

所有 $j\geqslant j_0$ 都成立的最小 j_0 为预周期(preperiod)。

例 4.2.1 令 $n=2$，在 F_2 上给定 2 级 LFSR 如图 4.2.2 所示。易知，此 2 级 LFSR 的反馈函数是 $a_{j+2}=a_{j+1}\oplus a_j$，反馈多项式是 $f(D)=1+D+D^2$，现给定初态 $(a_0,a_1)=(1,1)$。由式(4.2.3)可求得 $P(D)=1$(乘积 $f(D)\cdot a(D)$的小于 2 次的部分)。

计算 $P(D)/f(D)$的长除法的竖式如下：

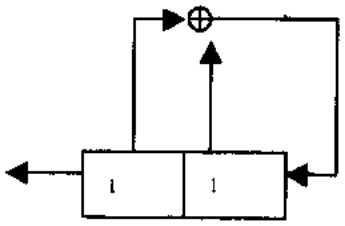


图 4.2.2 F_2 上的一个 2 级 LFSR

```
        1101…
   ┌──────
111 │ 100
    │ 111
    ├──────
      110
      111
    ├──────
      010
      000
    ├──────
      100
      …
```

根据式 (4.2.4)，我们就完全确定了此 LFSR 在给定初态(1,1)下的输出序列 $a=110110\cdots$。显然，a 的周期 $p(a)=3=2^2-1$。

F_2 上的 LFSR 产生的序列具有下述周期特性。

定理 4.2.1 设 n 是任意一个正整数，那么 F_2 上的任意 n 级 LFSR 产生的序列 a 都是终归周期的，且最小终归周期 $p(a)\leqslant 2^n-1$。

证明：对 n 级 LFSR，我们称 $\bar{a}_j=(a_j,a_{j+1},\cdots,a_{j+n-1})$ 为 LFSR 的第 j 个状态。当 LFSR 的第 j 个状态为全零状态即(0,0,…,0)时，LFSR 从第 $j+1$ 时刻起的输出全为零，显然该序列是终归周期为 1 的序列，其最小终归周期 $p(a)\leqslant 2^n-1$。当 LFSR 的任何时刻的状态都不是全零状态时，因为在 F_2 上共有 2^n-1 个非零 n 维向量，所以 LFSR 的状态集 $\{\bar{a}_j\mid 0\leqslant j\leqslant 2^n-1\}$中至少有两个相同。因此，存在 j 和 $i(0\leqslant i<j\leqslant 2^n-1)$使得 $\bar{a}_i=\bar{a}_j$，从而由 LFSR 的结构可知，LFSR 产生的序列 a 都是终归周期的，且最小终归周期 $p(a)\leqslant j-i\leqslant 2^n-1$。

由定理 4.2.1 可知，n 级 LFSR 产生的序列 a 的周期至多为 2^n-1，如果它的周期达到 2^n-1，我们就说这个序列是 n 级最大周期 LFSR 序列，简称 m-序列。把该 LFSR 称作最长 LFSR。

定理 4.2.2 设 LFSR 的反馈函数为式(4.2.1)，若 $c_n\neq 0$，则由该 LFSR 产生的序列 a 是周期序列。

证明：由定理 4.2.1 知，a 是终归周期的。设 $p(a)$是最小周期，j_0 是预周期，则 $a_{j+p(a)}=a_j$ 对所有 $j\geqslant j_0$ 成立。假如 $j_0\geqslant 1$，由于 $c_n\neq 0$，在式(4.2.1)中令 $j=j_0+p(a)-1$，则

$$a_{j_0+p(a)-1+n}=\sum_{i=1}^{n-1}c_i a_{j_0+p(a)-1+n-i}+a_{j_0+p(a)-1}$$

即

$$a_{j_0+p(a)-1}=a_{j_0+p(a)-1+n}+\sum_{i=1}^{n-1}c_i a_{j_0+p(a)-1+n-i}=a_{j_0-1+n}+\sum_{i=1}^{n-1}c_i a_{j_0-1+n-i}$$

在式(4.2.1)中再令 $j=j_0-1$，则可得

$$a_{j_0-1}=a_{j_0-1+n}+\sum_{i=1}^{n-1}c_i a_{j_0-1+n-i}$$

显然 $a_{j_0+p(a)-1}=a_{j_0-1}$，这与预周期的定义矛盾，故 $j_0=0$，即 a 为周期序列。

如果 n 级 LFSR 的反馈函数中的 $c_n\neq 0$，则称该 n 级 LFSR 是非退化的。由定理 4.2.2 可知，非退化的 LFSR 序列是周期的。

根据密码学的需要，对于 LFSR 主要考虑下面两个问题：

(1)如何利用级数尽可能小的 LFSR 产生周期长、统计特性好的序列；

(2)已知一个序列 a，如何构造一个尽可能短的 LFSR 来产生 a。

假定二元序列 $a\neq 0$，$p(a)=L$。显然，联接多项式为 $1+x^L$ 的 L 级 LFSR(该 LFSR 称作纯轮换移位寄存器(PCR))可以给出输出序列 a，联接多项式为 $1+x^{2L}$，$1+x^{3L}$，…的 LFSR 亦然。可见，能产生 a 的 LFSR 的联接多项式构成一个非空集合，记

$$J(a)=\{g(x)\mid g(x)\in F_2[x], g(D)a=0\}$$

设 $m_a(x)$ 是 $J(a)$ 中次数最低的非零多项式，则

$$J(a)=\{h(x)m_a(x)\mid h(x)\in F_2[x]\}$$

即 $J(a)$ 中的元素都是 $m_a(x)$ 的倍式。这是因为，如果 $g(x)\in J(a)$，当 $g(x)=0$ 时，显然 $g(x)$ 是 $m_a(x)$ 的倍式。当 $g(x)\neq 0$ 时，由 Euclidean 除法可得 $g(x)=h(x)m_a(x)+r(x)$，$\partial^0 r(x)<\partial^0 m_a(x)$。现在我们来说明 $r(x)=0$。由于 $g(D)a=0$，所以 $(h(D)m_a(D)+r(D))a=0$，即 $h(D)m_a(D)a+r(D)a=0$，而 $m_a(D)a=0$，故 $r(D)a=0$，这样 $r(x)\in J(a)$，而 $m_a(x)$ 是 $J(a)$ 中次数最低的非零多项式，所以只能 $r(x)=0$。从而 $g(x)=h(x)m_a(x)$。通常称 $m_a(x)$ 为周期序列 a 的极小多项式，也称极小联结式。显然，用 $m_a(x)$ 为联结多项式的 LFSR 产生 a 的方案是最优的。所以，需要判断当前用来产生 a 的 LFSR 的联接多项式 $f(x)$ 是否就是 $m_a(x)$。

由于序列的周期与多项式的周期或阶密切相关，我们现在来介绍多项式的周期的概念。

定义 4.2.2 设 $g(x)\in F_2[x]$，$\partial^0 g\geqslant 1$，$g(0)\neq 0$，使 $g(x)\mid(1-x^L)$ 成立的最小正整数 L 称为 $g(x)$ 的周期(又称阶或指数)，记为 $p(g)$。

我们先来说明一下定义 4.2.2 是合理的，即需说明对任何一个 $g(x)\in F_2[x]$，$\partial^0 g=m\geqslant 1$，$g(0)\neq 0$，存在正整数 L，使得 $g(x)\mid(1-x^L)$。考虑剩余类环 $F_2[x]/(f)$，即 $F_2[x]$ 中的多项式模 $f(x)$ 构成的环。在 $F_2[x]/(f)$ 中含有 2^m-1 个非零剩余类。2^m 个剩余类 $\{x^j+(f)\mid 0\leqslant j\leqslant 2^m-1\}$ 全部非零，故存在正整数 r 和 s，$0\leqslant r<s\leqslant 2^m-1$，使得 $x^s=x^r \bmod f(x)$。由于 x 与 $f(x)$ 互素，从而得出 $x^{s-r}\equiv 1 \bmod f(x)$，即 $f(x)\mid(x^{s-r}-1)$，$0<r-s\leqslant 2^m-1$。这也同时说明了多项式 $g(x)$ 的周期的上界为 2^m-1，其中 $\partial^0 g=m$。

定理 4.2.3 设 $m_a(x)$ 是非退化 LFSR 产生的序列 a 的极小多项式，则 $p(a)=p(m_a)$，即 a 的周期等于 $m_a(x)$ 的周期。

证明：由定义 4.2.2 可知，$m_a(x)\mid(1-x^{p(m_a)})$，故 $1-x^{p(m_a)}\in J(a)$，这说明序列 a 可以由 $p(m_a)$ 级纯轮换 LFSR 产生，因而 $p(a)\leqslant p(m_a)$。

另一方面，序列 a 显然可以由 $p(a)$ 级纯轮换 LFSR 产生，$1-x^{p(a)}\in J(a)$，故 $m_a(x)\mid(1-x^{p(a)})$；由定义 4.2.2 知，必有 $p(m_a)\leqslant p(a)$。

终上所述，$p(a)=p(m_a)$。

设 $f(x)\in F_2[x]$，$\partial^0 f\geqslant 1$，如果 $f(x)$ 不能表成 $F_2[x]$ 中两个次数不小于 1 的多项式的乘积，我们就说 $f(x)$ 在 F_2 上是不可约的；如果 $f(x)\neq x$ 是 F_2 上的一个 n 次不可约多项

式，并且 $f(x)$的周期 $p(f)=2^n-1$，我们就说 f 是 F_2 上的一个 n 次本原多项式。由定义 4.2.2 的合理性的论证可知，本原多项式的周期达到了多项式的周期的上界 2^n-1。

定理 4.2.4 设 LFSR 的联结多项式 $f(x)$是 F_2 上常数项为 1 的不可约多项式，则对于 LFSR 的任一输出序列 $a\neq 0$，有 $m_a(x)=f(x)$，并且 $p(a)=p(f)$。

证明：因为 $f(D)a=0$，所以 $f(x)\in J(a)$，从而有 $m_a(x)\mid f(x)$。但 $f(x)$是常数项为 1 的不可约多项式，故 $m_a(x)=f(x)$。再根据定理 4.2.3 可知，$p(a)=p(f)$。

下面给出 LFSR 产生 m-序列的充要条件。

定理 4.2.5 设 F_2 上 n 级 LFSR 的联结多项式为 $f(x)$，用 $G(f)$表示可由此 LFSR 产生的全部序列，则 $G(f)$中非零序列全是(n 级)m-序列的充要条件是 $f(x)$为 F_2 上的 n 次本原多项式。

证明：若 $f(x)$为 F_2 上的 n 次本原多项式，则由定理 4.2.4 可知，$G(f)$中的非零序列的周期全是 $p(f)=2^n-1$，因而 $G(f)$中的非零序列全是(n 级)m-序列。

反之，假定 $G(f)$中的非零序列全是(n 级)m-序列，我们只要能证明 $f(x)$是不可约多项式，则可利用定理 4.2.4 得到 $p(f)=2^n-1$，从而得知 $f(x)$是 n 次本原多项式。下面我们来证明 $f(x)$是不可约的。反设，有 $g(x),h(x)\in F_2[x]$，使得 $f(x)=g(x)h(x)$，$\partial^0 g\geqslant 1$，$\partial^0 h=k\geqslant 1$，则以 $h(x)$为联结多项式的 LFSR 产生的某个非零序列 $a\in G(f)$，这是因为 $h(D)a=0\Rightarrow f(D)a=0$，但此时 $p(a)\leqslant 2^k-1<2^n-1$，与定理的假设矛盾。

定理 4.2.5 表明，确定 F_2 上所有 m-序列的问题可转化为求 F_2 上所有本原多项式这个纯代数问题。代数中已经证明，F_2 上共有 $\varphi(2^n-1)/n$ 个 n 次本原多项式。并且，当 2^n-1 为素数时，F_2 上的每一个 n 次不可约多项式都是本原多项式[6,7]。

我们再来讨论 F_2 上的 m-序列的统计特性。

定义 4.2.3 设 $a=a_0a_1\cdots$是 F_2 上的周期为 $p(a)$的周期序列，称

$$C_a(\tau)=\frac{1}{p(a)}\sum_{k=0}^{p(a)-1}(-1)^{a_k+a_{k+\tau}}\qquad(0\leqslant\tau\leqslant p(a)-1)\qquad(4.2.5)$$

为周期序列 a 的(周期)自相关函数(auto-correlation function)，式中的 $\sum$ 表示按十进制求和。

定理 4.2.6[6,7,8] F_2 上的 n 级 m-序列 a，周期 $p(a)=2^n-1$，具有下述统计特性：

(1)在 a 的一个周期段中，1 出现 2^{n-1}次，0 出现 $2^{n-1}-1$ 次；

(2)将 a 的一个周期段首尾相接，其游程总数 $N=2^{n-1}$(所谓游程是指序列中同一符号的连续段，其前后为异种符号。例如…1$\underline{0001}$…与…0$\underline{1111}$0…分别称为一个长为 3 的 0-游程与一个长为 4 的 1-游程)。其中 0-游程与 1-游程的数目各半。当 $n>2$ 时，游程的分布如下($1\leqslant i\leqslant n-2$)：

1) 长为 i 的 0-游程 $N/2^{i+1}$个；

2) 长为 i 的 1-游程 $N/2^{i+1}$个；

3) 长为 $n-1$ 的 0-游程 1 个；

4) 长为 n 的 1-游程 1 个。

(3) a 的自相关函数是二值的，即

$$C_a(\tau)=\begin{cases}1 & \tau=0\\ -\dfrac{1}{p(a)} & 0<\tau\leq p-1\end{cases}$$

定理 4.2.6 的详细证明参见文献[6,7,8]等。

定理 4.2.6 中所描述的 m-序列的三条性质通常称作伪随机特性。定理 4.2.6 指出，m-序列具有周期大、统计特性类似于随机序列等优点。

至此，我们对第一个问题即如何利用级数尽可能小的 LFSR 产生周期大、统计特性好的序列作出了一个完美的回答。

下面我们来讨论第二个问题，即已知一个序列 a，如何构造一个尽可能短的 LFSR 来产生 a。Berlekamp-Massey 算法[9]（简称 B-M 算法）回答了这个问题。该算法是一个多项式时间的迭代算法，以 N 长二元序列

$$a_0,a_1,\cdots,a_{N-1} \tag{4.2.6}$$

为输入，输出产生序列式(4.2.6)的最短 LFSR 的联结多项式 $f_N(x)$及该 LFSR 的长度 l_N 的二元组$\langle f_N(x),l_N\rangle$。注意输出中的 l_N 是必要的，因为所求的 LFSR 可能是退化的而有 $\partial^0 f_N\neq l_N$。为叙述方便，我们约定，0 级 LFSR 是以 $f(x)=1$ 为联结多项式的 LFSR，并且 n 长($n=1,2,\ldots,N$)零序列：00…0 由而且只由 0 级 LFSR 产生。事实上，以 $f(x)=1$ 为联结多项式的递归关系式是

$$a_k=0,k=0,1,\cdots,n-1$$

因此，这一约定是合理的。

现在我们来描述著名的 B-M 算法[9]。

B-M 算法

输入：$N;a_0,a_1,a_2,\cdots,a_{N-1}$。

步骤 1：$n\leftarrow 0,\langle f_0(x),l_0\rangle\leftarrow\langle 1,0\rangle$。

步骤 2：计算 $d_n=f_n(D)a_n$；其中 D 为延迟算子。

(1) 当 $d_n=0$ 时

$$\langle f_{n+1}(x),l_{n+1}\rangle\leftarrow\langle f_n(x),l_n\rangle,\text{转步骤 3}$$

(2) 当 $d_n=1$，且 $l_0=l_1=\cdots=l_n=0$ 时

$$\langle f_{n+1}(x),l_{n+1}\rangle\leftarrow\langle 1+x^{n+1},n+1\rangle,\text{转步骤 3}$$

(3) 当 $d_n=1$，且 $l_m<l_{m-1}=l_{m+2}=\cdots=l_n(m<n)$时

$$\langle f_{n+1}(x),l_{n+1}\rangle\leftarrow\langle f_n(x)+x^{n-m}f_m(x),\max\{l_n,n+1-l_n\}\rangle$$

步骤 3：若 $n<N-1$，$n\leftarrow n+1$ 转步骤 2。

输出：$\langle f_N(x),l_N\rangle$。

算法中的 d_n 称为第 n 步离差(discrepancy)。易知，B-M 算法的时间复杂性为 $O(N^2)$，空间复杂性为 $O(N)$。

定理 4.2.7 使用 B-M 算法，以 N 长二元序列式(4.2.6)为输入，得到输出$\langle f_N(x),l_N\rangle$。则：

(1) 以 $f_N(x)$为联结多项式，长为 l_N 的 LFSR 是产生二元序列式(4.2.6)的最短 LFSR。

(2) 当且仅当 $l_N\leqslant N/2$ 时，产生 N 长二元序列式(4.2.6)的最短 LFSR 是唯一的。

(3) 当 $l_N<N/2$ 时，迭代至 $2l_N$ 步已有$\langle f_{2l_N}(x),l_{2l_N}\rangle=\langle f_N(x),l_N\rangle$ 。

定理 4.2.7 的证明参见文献[6,7]。B-M 算法的原始形式参见文献[10]。文献[6]中描述的是一般的有限域上的 B-M 算法。我们最关心的是如何求一个给定周期序列 a 的极小联结式 $m_a(x)$。用 B-M 算法求 $m_a(x)$，有如下结论。

定理 4.2.8 设二元序列 a：

$$a_0,a_1,\cdots,a_{p(a)},\cdots \tag{4.2.7}$$

是周期为 $p(a)$的非零序列。那么

(1) 用B-M 算法求得的$\langle f_{2p(a)}(x),l_{2p(a)}\rangle$就是产生 a 的唯一的最短 LFSR，即 $m_a(x)=f_{2p(a)}(x)$，此时，$\partial^0 f_{2p(a)}(x)=l_{2p(a)}$。

(2) 记 $n=l_{2p(a)}$，则$\langle f_{2n}(x),l_{2n}\rangle=\langle f_{2p(a)}(x),l_{2p(a)}\rangle$，即 $m_a(x)=f_{2n}(x)$。特别地，对于 $p(a)=2^n-1$ 的 m-序列 a，$m_a(x)=f_{2n}(x)$。

定理 4.2.8 的证明可在文献[6,7]中找到。定理 4.2.8 表明，如果已知 a 是 n 级 m-序列，则利用 a 的前 $2n$ 位(或任意一个长为 $2n$ 的截段)，用 B-M 算法求得的 $f_{2n}(x)$就是要找的那个极小的联结式 $m_a(x)$。当然它一定是一个 n 次本原多项式。对于适当大的 n，显然 $2n\ll p(a)=2^n-1$。所以，用 B-M 算法求 $m_a(x)$的时间复杂性仅为 $O(n^2)$，它远远小于 $O(p_{(a)}^2)$。于是，我们得出如下结论：m-序列生成器适宜用在密钥流生成器的驱动部分，完成将实际密钥 k(或 k 的某一部分)扩展成周期大、统计特性好的驱动序列的任务，而不能单独用作密钥源。

例 4.2.2 设 F_2 上的一个长为 7 的序列 a 为

$$a_0=0,a_1=0,a_2=1,a_3=1,a_4=1,a_5=0,a_6=1$$

求产生该序列的一个最短 LFSR。

由 B-M 算法，我们有

$$\begin{aligned}
&\langle f_0(x),l_0\rangle=\langle 1,0\rangle && d_0=0\\
&\langle f_1(x),l_1\rangle=\langle 1,0\rangle && d_1=0\\
&\langle f_2(x),l_2\rangle=\langle 1,0\rangle && d_2=0\\
&\langle f_3(x),l_3\rangle=\langle 1+x^3,3\rangle && d_3=1\\
&\langle f_4(x),l_4\rangle=\langle 1+x+x^3,3\rangle && d_4=0\\
&\langle f_5(x),l_5\rangle=\langle f_4(x),l_4\rangle && d_5=0\\
&\langle f_6(x),l_6\rangle=\langle f_5(x),l_5\rangle && d_6=0\\
&\langle f_7(x),l_7\rangle=\langle f_6(x),l_6\rangle
\end{aligned}$$

这表明，$\langle 1+x+x^3,3\rangle$为产生所给序列的一个最短 LFSR(参见图 4.2.3)。

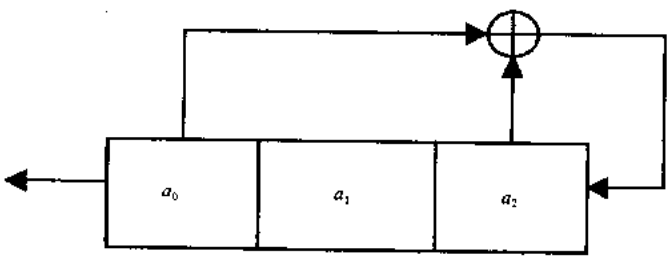


图 4.2.3 产生序列 a 的 LFSR

本节简要地介绍了有关 LFSR 的一些基本事实。想全面了解 LFSR 的读者可参阅文献[5,6,7,8,11]等。这些文献中对非线性反馈移位寄存器也作了比较详细的论述。

4.3 随机性、线性复杂度和 Blahut 定理

为了度量周期序列的随机性，Golomb[8]提出了下列三条标准(以 F_2 上的周期序列为例)：(1)一个周期中 0 与 1 的数目基本平衡；(2)一个周期中相同长度的 0-游程与 1-游程的数目基本平衡；(3)周期自相关函数为二值函数。通常称符合上述三条标准或部分标准的序列(诸如 m-序列，二次剩余序列等)为伪随机序列或伪噪声序列(简记为 PN 序列)。然而，满足上述三条标准的周期序列并不能满足我们对密钥流序列在安全性方面的要求，这种安全性是寓于随机性之中的。一个二元随机序列 $a_0,a_1,a_2,\cdots$ 可视作一个二元对称信源(BSS)的输出(也可用掷硬币的方式产生)，它的随机性包含着当前输出位 a_n 对以前输出段 $a_0,a_1,\cdots,a_{n-1}$ 的完全独立性，即 $H(a_n|a_0,a_1,\cdots,a_{n-1})=H(a_n)$，对一切 $n\geqslant 1$。故在已知 $a_0,a_1,\cdots,a_{n-1}$ 的条件下，a_n 仍是不可预测的。而对于 n 级 m 序列，由上节讨论知，只要得知其前 $2n$ 位，即能以不太大的代价 $O(n^2)$ 预测该序列的任一位的取值。然而，要移植关于 BSS 的不可预测性这一概念并不是一件容易的事情，因为 BSS 提供的无限长随机序列与我们用作密钥流的周期序列之间存在着本质的差别。

早在 60 年代，Solomonov[12]和 Kolmogorov[13]就基于随机序列的不可预测性提出用所谓的“无定型性”(Pattern lessness)来描述有限长序列的随机性，规定一个有限长序列的无定型性由产生该序列的最短图林机程序长度来度量。另一种度量有限长或周期序列的随机性的方法是由 Lempel 和 Ziv[14]建议的所谓“线性复杂度”方法，规定用产生该序列的最短的 LFSR 的长度来度量，这种方法实际上衡量了序列的线性不可预测性。

为了叙述方便起见，我们仅限制在二元域 F_2 上讨论。

定义 4.3.1 F_2 上的一个有限长或半无限长序列 a 的线性复杂度(linear complexity)定义为

$$L(a)=\min\{n|\text{存在 } F_2 \text{ 上的 } n \text{ 级 LFSR 产生 } a\}$$

约定，$L(0)=0$。

从理论上讲，任一确定的序列 a 的线性复杂度 $L(a)$ 都可以利用 B-M 算法求得。特别地，对周期为 2^n 的二元序列有一个更有效的算法，参见文献[15,16]。

序列的线性复杂度的基本性质：设 a 和 b 是 F_2 上的两个序列。

(1) 对任何 $n\geqslant 1$，序列 $a^n=a_0a_1\cdots a_{n-1}$ 的线性复杂度 $L(a^n)$ 满足 $0\leqslant L(a^n)\leqslant n$；

(2) $L(a^n)=0$，当且仅当 a^n 是长为 n 的零序列；

(3) $L(a^n)=n$，当且仅当 $a^n=0,0,\cdots,0,1$；

(4) 如果 a 是周期为 N 的序列，则 $L(a)\leqslant N$；

(5) $L(a\oplus b)\leqslant L(a)+L(b)$，这里 $a\oplus b$ 表示 a 和 b 的逐比特异或。

线性复杂度小的序列肯定不能用作密钥流序列。那么，是否线性复杂度大的序列就一定可以用作密钥流序列呢？答案是否定的。一个明显的例子是周期序列 $a=\underbrace{00\cdots01}_{p\text{长}}\,00\cdots$。$a$ 的线性复杂度 $L(a)=p$，但 a 显然不能用作密钥流。出现这种情况的主要原因是线性复

杂度的定义将预测的方式严格局限于线性方式。然而，对有限长或周期序列 a，$L(a)$是目前能明确计算出的一个不可多得的指标，它仍然是度量密钥流序列的随机性的一个重要指标。人们围绕线性复杂度进行了一系列研究，也给出了许多补救线性复杂度的不足的措施，并提出了各种各样的度量有限长或周期序列的随机性的指标，诸如球复杂度(sphere complexity)或 d-复杂度(d-complexity)[16,17]、跃复杂度(jump complexity)[18]、线性复杂度轮廓(linear complexity profiles)[1]、非线性复杂度[19,20,21]等。

下面我们将讨论有限长或周期序列的线性复杂度轮廓，至于其它问题，感兴趣的读者请参阅有关文献。

首先，我们来考虑随机序列或 BSS 输出序列的一个 N 长截段 $a^N=a_0a_1\cdots a_{N-1}$的线性复杂度的期望值和方差。

显然，a^N 给定一个包含 2^N 个样本序列的有限概率空间 $\Omega=\Omega_N$，其中每一个样本序列出现的概率是相等的。我们用 b^N 记 Ω 中的某个样本序列，则概率 $p(b^N)=1/2^N$。于是 a^N 的线性复杂度的期望值(涉及期望值与方差计算的求和、相乘等运算按十进制理解)

$$E[L(a^N)]=2^{-N}\sum_{b^N\in\Omega}L(b^N)=2^{-N}\sum_{l=1}^{N}l\cdot\sigma_N(l) \tag{4.3.1}$$

其中 $\sigma_N(l)$表示 Ω 中线性复杂度为 l 的序列的个数。文献[1]中已给出 $\sigma_N(l)$的分布，其分布为

$$\sigma_N(l)=\begin{cases}2^{\min(2N-2l,2l-1)} & N\geqslant l>0\\ 1 & l=0\end{cases} \tag{4.3.2}$$

将式(4.3.2)代入式(4.3.1)整理后可得

$$E[L(a^N)]=2^{-N}\Big[\sum_{l=1}^{[N/2]}l\cdot 2^{2l-1}+\sum_{l=[N/2]+1}^{N}l\cdot 2^{2N-2l}\Big] \tag{4.3.3}$$

由高等数学的知识，易求出

$$\sum_{j=1}^{m}j\cdot 2^{2j-1}=\frac{m+1}{3}\cdot 2^{2m+1}-\frac{2}{9}(2^{2m+2}-1) \tag{4.3.4}$$

利用式(4.3.4)和式(4.3.3)可得

$$E[L(a^N)]=\frac{N}{2}+\frac{4+N(\mathrm{mod}2)}{18}-2^{-N}\left(\frac{N}{3}+\frac{2}{9}\right) \tag{4.3.5}$$

式(4.3.5)给出了由 N 个独立的、均匀分布的二元随机变量给出的序列 $a^N=a_0a_1\cdots a_{N-1}$的线性复杂度的期望值。

下面再研究 $L(a^N)$的方差

$$\mathrm{Var}[L(a^N)]=E[L^2(a^N)]-(E[L(a^N)])^2 \tag{4.3.6}$$

类似于 $E[L(a^N)]$的讨论可得

$$E[L^2(a^N)]=\frac{N^2}{4}+\frac{4+N(\mathrm{mod}2)}{18}N+\frac{40+N(\mathrm{mod}2)}{36}-2^{-N}\left(\frac{N^2}{3}+\frac{8N}{9}+\frac{10}{9}\right) \tag{4.3.7}$$

将式(4.3.5)和式(4.3.7)代入式(4.3.6)可得

$$\mathrm{Var}[L(a^N)]=\frac{86}{81}-2^{-N}\times\left(\frac{14-N(\mathrm{mod}2)}{27}N+\frac{82-2(N(\mathrm{mod}2))}{81}\right)-2^{-2N}\left(\frac{N^2}{9}+\frac{4N}{27}+\frac{4}{81}\right) \tag{4.3.8}$$

式(4.3.8)给出了由 N 个独立的、均匀分布的二元随机变量给出的序列 $a^N=a_0a_1\cdots a_{N-1}$ 的线性复杂度的方差。

由式(4.3.5)和式(4.3.8)知

$$E[L(a^N)]=\frac{N}{2}+\frac{1}{4}+\frac{(-1)^N}{36}+O(N2^{-N}) \tag{4.3.9}$$

$$\mathrm{Var}[L(a^N)]=\frac{86}{81}+O(N2^{-N}) \tag{4.3.10}$$

式(4.3.9)和(4.3.10)这两个结果的一个最直观的解释是以获取"随机序列"为目的，所选 N 长二元序列 b^N 的线性复杂度 $L(b^N)$ 的最佳值是$[N/2]$或$[N/2]\pm1$，因为此时 b^N 有较大可能"像"BSS 输出的大多数 N 长截段。

定义 4.3.2 设 $a^N=a_0a_1\cdots a_{N-1}$ 是 F_2 上的一个 N 长序列，$L_i=L(a^i)$ 是子序列 $a^i=a_0a_1\cdots a_{i-1}$ 的线性复杂度，则称序列 $L_1,L_2,\cdots,L_N$ 是序列 a^N 的线性复杂度轮廓(LCP)。

式(4.3.9)表明，二元随机序列的线性复杂度轮廓近似地按 $N/2$ 变化。

文献[1,22]中的结果表明，周期为 T 的所有二元序列的线性复杂度轮廓均近似地按周期 T 变化。

线性复杂度轮廓十分自然地提供了一种统计测试方法：好的滚动密钥生成器的线性复杂度轮廓和随机序列的线性复杂度轮廓应是"不可区分的"[1]。

原则上，序列 a 的线性复杂度 $L(a)$ 可由 B-M 算法求出，但在某些应用场合，使用 B-M 算法并不方便，有时人们只需要估计一个序列的线性复杂度的上、下界，而无需具体求出，这就需要采取别的方法，诸如下面将要介绍的用周期序列的不可约有理形求线性复杂度的方法就是估计线性复杂度的一种很有用的方法。文献[16]中主要用这种方法来估计线性复杂度的界。

对 F_2 上的周期为 N(未必最小)的周期序列 $a=a_0a_1a_2\cdots a_{N-1}\cdots$，它的 D-变换为

$$\begin{aligned}a(D)=&a_0+a_1D+\cdots+a_{N-1}D^{N-1}+a_0D^N\\&+a_1D^{N+1}+\cdots+a_{N-1}D^{2N-1}\cdots=\frac{a^N(D)}{1-D^N}\end{aligned} \tag{4.3.11}$$

其中

$$a^N(D)=a_0+a_1D+\cdots+a_{N-1}D^{N-1}$$

设 a 的极小联结式为 $m_a(x)$，由式(4.2.4)可知

$$a(D)=\frac{P(D)}{m_a(D)}\qquad \partial^0P<\partial^0m_a \tag{4.3.12}$$

但 $1-x^N\in J(a)$，所以 $m_a(x)\mid(1-x^N)$，即存在 $d(x)\in F_2[x]$ 使得 $m_a(x)d(x)=1-x^N$。因此由式(4.3.11)可知

$$a(D)=\frac{P(D)}{m_a(D)}=\frac{P(D)d(D)}{m_a(D)d(D)}=\frac{P(D)d(D)}{1-D^N} \tag{4.3.13}$$

由式(4.3.11)和式(4.3.13)可知，$P(D)d(D)=a(D)(1-D^N)=a^N(D)$，从而得知 $m_a(D)$ 也是通过对式(4.3.11)的右端约化得来的。这就证明了

$$m_a(x)=(1-x^N)/\gcd(a^N(x),1-x^N) \tag{4.3.14}$$

式(4.3.14)表明，周期序列的不可约有理分式表示是唯一的，且有理分式的分母一定是极小联结式。式(4.3.14)不仅仅是一种形式表示，它表明可用 Euclidean 算法来确定产

生周期序列的最短 LFSR(等价地,确定序列的极小多项式)。这自然使人们联想到 B-M 算法和 Euclidean 算法应有某种联系。关于这种联系的探讨请参阅文献[23,24,25]。

在这里我们顺便谈一谈 $F_2\langle D\rangle$ 中的乘法逆元素。设 $B(D),C(D)\in F_2\langle D\rangle$,如果 $B(D)C(D)=1$,则称 $B(D)$ 和 $C(D)$ 互为乘法逆。

设 $B(D)=\sum_{n=0}^{\infty}b_nD^n\in F_2\langle D\rangle$,$B(D)$ 在 $F_2\langle D\rangle$ 中有乘法逆元素,当且仅当 $b_0\neq 0$,即 $b_0=1$。事实上,如果 $B(D)$ 在 $F_2\langle D\rangle$ 中有乘法逆元素,由定义易知,$b_0\neq 0$。反之,如果 $b_0\neq 0$,设 $C(D)=\sum_{n=0}^{\infty}c_nD^n\in F_2\langle D\rangle$,如果 $B(D)C(D)=1$,则下述一系列方程成立:

$$\begin{aligned}
&b_0c_0=1\\
&b_0c_1+b_1c_0=0\\
&b_0c_2+b_1c_1+b_2c_0=0\\
&\quad\vdots\\
&b_0c_n+b_1c_{n-1}+\cdots+b_nc_0=0
\end{aligned}$$

由于 $b_0\neq 0$(此时 $b_0=1$),所以由第一个方程唯一确定 $c_0=b_0^{-1}=1$,再由第二个方程唯一确定出 c_1,一般而言,利用第一个方程和逆归关系式

$$c_n=-b_0^{-1}\sum_{k=1}^{n}b_kc_{n-k}=\sum_{k=1}^{n}b_kc_{n-k}\qquad(n=1,2,\cdots)$$

求出 c_n,所得形式幂级数 $C(x)$ 是 $B(x)$ 的乘法逆。

序列的线性复杂度和矩阵的秩有内在的联系。下面的定理刻划了这种关系[26]。下面我们在一般的有限域 F_q 上进行讨论。

定理 4.3.1 设 $a^n=a_0a_1\cdots a_{n-1}$ 是 F_q 上的一个 n 长序列,则 a^n 的线性复杂度等于下面矩阵的最小秩:

$$G_{a^n}=\begin{bmatrix}
a_0 & a_1 & a_2 & \cdots & a_{n-3} & a_{n-2} & a_{n-1}\\
a_1 & a_2 & a_3 & \cdots & a_{n-2} & a_{n-1} & *\\
a_2 & a_3 & a_4 & \cdots & a_{n-1} & * & *\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
a_{n-1} & * & * & \cdots & * & * & *
\end{bmatrix}\tag{4.3.15}$$

这里 $*$ 为 F_q 中任意元素,且最小值是对这些任意元素赋所有可能值而求的。

证明:我们约定 G_{a^n} 最上面的行为第 0 行。如果 $L(a^n)=L$,则由线性复杂度的定义可知,对于 $L\leqslant i\leqslant n-1$,只要适当地选择这些任意元素,$G_{a^n}$ 的第 i 行可表示成前面 L 行的线性组合。因此,G_{a^n} 的最小秩不超过 L。但又由线性复杂度的定义可知,L 是产生 a^n 的最短的 LFSR 的级数,因而 L 是第 L 行的前 $n-L$ 个分量能够表示成前面行的对应分量线性组合的最小整数。这说明前 L 行线性独立,从而 G_{a^n} 的最小秩不小于 L,故 G_{a^n} 的最小秩等于 L。

定理 4.3.2 设 $a^\infty=a_0a_1\cdots a_{N-1}\cdots$ 是 F_q 上的一个周期为 N 的序列,则 a^∞ 的线性复杂度等于下面循环矩阵的秩:

$$M(a^\infty)=\begin{bmatrix}
a_0 & a_1 & \cdots & a_{N-1}\\
a_1 & a_2 & \cdots & a_0\\
\vdots & \vdots & \cdots & \vdots\\
a_{N-1} & a_0 & \cdots & a_{N-2}
\end{bmatrix}\tag{4.3.16}$$

定理 4.3.2 的证明类似于定理 4.3.1 的证明，故略。

有限序列的汉明(Hamming)重量与另一个相关序列的线性复杂度之间的关系由 Blahut 给出[27]，文献[26]将 Blahut 的结果应用到编码中去研究线性码的距离，文献[16]将 Blahut 的结果应用于研究线性复杂度稳定性指标之间的关系，文献[28,29]将 Blahut 的结果应用于研究非线性滤波序列。可见，Blahut 定理在编码和密码的研究中起着重要的作用。下面我们来介绍 Blahut 定理。在介绍这个定理之前，我们先来介绍域 F_q 上的离散傅里叶(Fourier)变换(简记为 DFT)。

设 α 是 F_q 的某一扩域 F_{q^m} 中的一个 N 阶元素。一个 F_q 上的序列 $a^N=a_0a_1\cdots a_{N-1}$(也称为“时-域”(time-domain)序列)的 DFT 定义为一个 F_{q^m} 上的序列 $A^N=A_0A_1\cdots A_{N-1}$(也称为“频-域”(frequency-domain)序列)，其中

$$A_i=\sum_{j=0}^{N-1}a_j\alpha^{ij}\qquad i=0,1,\cdots,N-1 \tag{4.3.17}$$

逆 DFT 为

$$a_j=\frac{1}{N^*}\sum_{i=0}^{N-1}A_i\alpha^{-ij}\qquad j=0,1,\cdots,N-1 \tag{4.3.18}$$

其中 $N^*=N\bmod p$，p 是 F_q 的特征。

令

$$F=\begin{pmatrix}1 & 1 & 1 & \cdots & 1\\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{N-1}\\ \vdots & \vdots & \vdots & \cdots & \vdots\\ 1 & \alpha^{N-1} & \alpha^{2(N-1)} & \cdots & \alpha^{(N-1)(N-1)}\end{pmatrix}$$

则 A^N 可表示为

$$A^N=a^NF \tag{4.3.19}$$

a^N 可表示为

$$a^N=\frac{1}{N^*}A^NF^{-1} \tag{4.3.20}$$

这里

$$F^{-1}=\begin{pmatrix}1 & 1 & 1 & \cdots & 1\\ 1 & \alpha^{-1} & \alpha^{-2} & \cdots & \alpha^{-(N-1)}\\ \vdots & \vdots & \vdots & \cdots & \vdots\\ 1 & \alpha^{-(N-1)} & \alpha^{-2(N-1)} & \cdots & \alpha^{-(N-1)(N-1)}\end{pmatrix}$$

如果对所有的 i，我们按式(4.3.17)来定义 A_i，则有

$$A_{i+N}=\sum_{j=0}^{N-1}a_j\alpha^{(i+N)j}=\sum_{j=0}^{N-1}a_j\alpha^{ij}=A_i$$

于是，V^∞是周期为 N 的序列。如果对所有的 i，我们按式(4.3.18)来定义 a_i，则同样可以证明 a^∞是周期为 N 的序列。因此，可将 a^N 和 A^N 看成是由式(4.3.18)和式(4.3.17)分别定义的周期序列 a^∞和 A^∞的第一个周期段。

由定理 4.3.2 可知，a^∞的线性复杂度等于下面循环矩阵的秩，即

$$M(a^{\infty})=\begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \\ a_1 & a_2 & \cdots & a_0 \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-1} & a_0 & \cdots & a_{N-2} \end{bmatrix}$$

$M(a^{\infty})$可写成

$$M(a^{\infty})=\frac{1}{N^*}FD_AF^{-1}$$

其中

$$D_A=\begin{pmatrix} A_0 & & & \\ & A_1 & & \\ & & \ddots & \\ & & & A_{N-1} \end{pmatrix}$$

因为 F 和 F^{-1}互为逆矩阵，故秩 $M(a^{\infty})=$秩 $D_A=W_H(A^N)$，$W_H(A^N)$表示 A^N 的汉明(Hamming)重量。类似地可证明，$L(A^{\infty})=W_H(a^N)$。

定理 4.3.3(Blahut 定理) 周期为 N 的半无限"时域"序列 a^{∞}的线性复杂度等于有限长"频域" 序列 A^N 的汉明重量，即 $L(a^{\infty})=W_H(A^N)$，其中 A^N 是 a^N 的 DFT。类似地，周期为 N 的半无限序列 A^{∞}的线性复杂度等于有限长序列 a^N 的汉明重量，即 $L(A^{\infty})=W_H(a^N)$其中 a^N 是 A^N 的 DFT。

Blahut 定理表明，一个周期半无限序列的线性复杂度等于它的一个周期的 DFT 的汉明重量。

值得注意的是，在有限域上，不是对所有的 N 都存在离散傅里叶变换，因为不是每一个数都可以作为元素的阶。但是，对大多数的目的来说，这已经足够了。如果 m 是 q^m-1 能够被 N 除尽的最小整数，那么在 F_q 上就有长为 N 的有限域离散傅里叶变换，其分量是在 F_{q^m}上。不幸的是，对于某些值，虽然变换是存在的，但是它存在于很大的扩域中，因而不一定实用。

例 4.3.1 设 $a^{\infty}=0111010\cdots$是 F_2 上的一个周期为 7 的序列，α 是域 F_{2^3}的一个本原元，即 α 的阶为 7。$a^7=0111010$ 的 DFT 为 $A^7=0\alpha\alpha^2 0\alpha^4 00$，由 Blahut 定理可知 $L(a^{\infty})=W_H(A^7)=3$。反之，$L(A^{\infty})=W_H(a^7)=4$。

4.4 布尔函数的非线性准则

在研究同步流密码生成器时，Ruppel 将这类生成器分成两部分，即驱动部分和非线性组合部分。驱动部分的任务是控制存储器的状态转移，负责提供若干供组合部分使用的周期大、统计特性好的序列，而非线性组合部分的任务是将驱动部分所提供的序列组合成满足某些要求的好的密钥流序列。由于 LFSR 序列的理论已经成熟，特别是 m-序列具有良好的伪随机性，所以驱动部分的设计比较容易。于是非线性组合部分的设计已成为密钥流生成器设计的重点和难点。为了能使密钥流生成器抵抗已知的若干种攻击方法，在功能上人们已对非线性组合部分提出了多项要求。自然，随着人们对问题研究的深入，特别是某种新的攻击方法的出现，还可能对非线性组合部分提出一些新的、更深刻、更明确的要

求。非线性组合部分可由逻辑函数来实现，因此，非线性组合部分的研究可归纳为逻辑函数的研究。可见，对非线性组合部分的要求也就是对逻辑函数的要求，因此研究逻辑函数的密码学性质是必要的，也是有意义的。目前关于逻辑函数的密码学性质的研究主要包括以下几个方面：

(1) 非线性次数；

(2) 非线性度(相关度)；

(3) 线性结构；

(4) 退化性；

(5) 相关免疫性；

(6) 严格雪崩准则和扩散准则。

通常也将逻辑函数的这些密码学性质称为逻辑函数的非线性准则。

本节我们主要讨论二值逻辑函数即布尔函数的非线性准则，关于多值逻辑函数的非线性准则的研究，感兴趣的读者可参阅文献[30,31,32,33,34,35,36,37,38,39]。

4.4.1 布尔函数的表示和 Walsh 谱

n 个变元的布尔函数 $f(x)$是从 F_2^n 到 F_2 的一个函数或映射，一般记为 $f(x):F_2^n\to F_2$，它有多种表示形式。本节我们主要来介绍布尔函数 $f(x):F_2^n\to F_2$ 的四种表示形式，即真值表表示、小项表示、多项式表示和 Walsh 谱表示。

4.4.1.1 真值表表示

一个 n 元布尔函数 $f(x):F_2^n\to F_2$ 是否给定，关键在于该函数之值是否对于每一组自变量 $x=(x_1,x_2,\cdots,x_n)$均已确定。如果把每一组自变量$(x_1,x_2,\cdots,x_n)$与其所对应的函数值全部列成表格，这种表格就叫做布尔函数的真值表。习惯上总是按二进制表示 $x_1,x_2,\cdots,x_n$ 之值递增的顺序由上到下排列真值表(视 x_n 为最低位)。在此约定下，将表中函数值构成的行矢量记为 f，称为函数值矢量。f 的汉明重量称为 f 的重量，记为 $W_H(f)$，即 $f(x)=1$的 x 的个数。特别地，当 $W_H(f)=2^{n-1}$时，我们说 f 是平衡布尔函数。

表 4.4.1 布尔函数真值表表示实例

x_1	x_2	$f(x)=f(x_1,x_2)$
0	0	1
0	1	1
1	0	1
1	1	0

例 4.4.1 设 $f(x):F_2^2\to F_2$，其真值表如表 4.4.1 所示。表 4.4.1 所表示的函数 f 的函数值矢量 $\boldsymbol{f}=(1,1,1,0)$，重量为 $W_H(f)=3$。

4.4.1.2 小项表示

对于 $x_i,c_i\in F_2$，约定 $x_i^1=x_i,x_i^0=\bar{x}_i=1+x_i$，于是

$$x_i^{c_i}=\begin{cases}1, x_i=c_i\\ 0, x_i\neq c_i\end{cases}$$

设整数 $c(0\leqslant c\leqslant 2^n-1)$的二进制表示是 $c_1c_2\cdots c_n$，约定 $x^c=x_1^{c_1}x_2^{c_2}\cdots x_n^{c_n}$，它具有下述“正交性”：

$$x_1^{c_1}x_2^{c_2}\cdots x_n^{c_n}=\begin{cases}1,(x_1,x_2,\cdots,x_n)=(c_1,c_2,\cdots,c_n)\\ 0,(x_1,x_2,\cdots,x_n)\neq(c_1,c_2,\cdots,c_n)\end{cases}\tag{4.4.1}$$

由此可得到

$$f(x)=\sum_{c=0}^{2^n-1}f(c_1,c_2,\cdots,c_n)x_1^{c_1}x_2^{c_2}\cdots x_n^{c_n} \tag{4.4.2}$$

式(4.4.2)称为 f 的小项表示，每一个被加项 $f(c_1,c_2,\cdots,c_n)x_1^{c_1}x_2^{c_2}\cdots x_n^{c_n}$ 称为一个小项。其中求和符号“$\sum$”是指在 F_2 上的求和 。

例 4.4.2 表 4.1.1 表示的布尔函数 $f(x):F_2^n\to F_2$ 的小项表示为

$$\begin{aligned}f(x_1,x_2)&=1\cdot x_1^0x_2^0+1\cdot x_1^0x_2^1+1\cdot x_1^1x_2^0+0\cdot x_1^1x_2^1\\&=\bar{x}_1\bar{x}_2+\bar{x}_1x_2+x_1\bar{x}_2\end{aligned} \tag{4.4.3}$$

4.4.1.3 *多项式表示*

式(4.4.3)可以变形为 F_2 上的多项式

$$f(x_1,x_2)=(x_1+1)(x_2+1)+(x_1+1)x_2+x_1(x_2+1)=1+x_1x_2$$

这就得到了布尔函数 $f(x_1,x_2)$的多项式表示。一般地，将 $\bar{x}_i=1+x_i$ 代入式(4.4.2)并注意到 $x_ix_i=x_i$，$x_ix_j=x_jx_i$，利用分配律并且进行同类项合并，便可使该式化为变量 $x_1,x_2,\cdots,x_n$ 的一些单项式 $x_{i_1},x_{i_2},\cdots,x_{i_r}$ 之模 2 和，即

$$f(x_1,x_2,\cdots,x_n)=a_0+\sum_{r=1}^{n}\sum_{1\leqslant i_1<i_2<\cdots<i_r\leqslant n}a_{i_1i_2\cdots i_r}x_{i_1}x_{i_2}\cdots x_{i_r} \tag{4.4.4}$$

$a_0,a_{i_1i_2\cdots i_r}\in F_2$。称式(4.4.4)为 $f(x)$的多项式表示。

常将式(4.4.4)按变元升幂及下标的字典序写出

$$\begin{aligned}f(x)=a_0&+a_1x_1+a_2x_2+\cdots+a_nx_n+a_{12}x_1x_2\\&+\cdots+a_{n-1,n}x_{n-1}x_n+\cdots+a_{12\cdots n}x_1x_2\cdots x_n\end{aligned} \tag{4.4.5}$$

称式(4.4.5)为 $f(x)$的代数正规型。任一确定的 n 个变元的布尔函数 $f(x)$的代数正规型式(4.4.5)是唯一的。一个乘积项(也称单项式)$x_{i_1}x_{i_2}\cdots x_{i_r}$ 的次数定义为 r，非零常数项的次数定义为 0，0 的次数定义为 $-\infty$。布尔函数 f 的次数定义为 f 的代数正规型中具有非零系数的乘积项中的最大次数，记为 $\partial^0 f$。当 $\partial^0 f=1$ 时，称 $f(x)$为线性布尔函数；当 $\partial^0 f\geqslant 2$ 时，称 $f(x)$为非线性布尔函数。

我们看一下 $f(x)$的最高次项 $x_1x_2\cdots x_n$ 出现的充要条件，易知 $a_{12\cdots n}=\left(\sum_{x\in F_2^n}f(x)\right)\bmod 2$，这里“$\sum$”是实数求和。则 $a_{12\cdots n}=W_H(f)\bmod 2$，从而 $a_{12\cdots n}=1$，当且仅当 $W_H(f)$为奇数。由此可见，当 $W_H(f)$为偶数时，$a_{12\cdots n}=0$，即最高次项不出现。特别地，平衡布尔函数无最高次项 $x_1x_2\cdots x_n$。

式(4.4.4)中的单项式 $x_{i_1}x_{i_2}\cdots x_{i_r}$ 可以改写成更规范的形式 $R_k(x)=x_1^{k_1}x_2^{k_2}\cdots x_n^{k_n}$，式中 $k_1k_2\cdots k_n$ 是 $k(0\leqslant k\leqslant 2^n-1)$的二进制表示，此时约定

$$x_i^{k_i}=\begin{cases}x_i & k_i=1\\ 1 & k_i=0\end{cases} \tag{4.4.6}$$

若将相应系数记为 $f(k)$，则式(4.4.4)可改写成

$$f(x)=\sum_{k=0}^{2^n-1}f(k)R_k(x) \tag{4.4.7}$$

称式(4.4.7)为 $f(x)$的 Reed-Muller 谱表示(简称 Reed-Muller 谱)。记

$$\hat{f}(\boldsymbol{k}) = (f(0), f(1), \cdots, f(2^n - 1))$$

式(4.4.7)中的系数 $f(0), f(1), \cdots, f(2^n-1)$ 可由下式计算出：

$$\begin{aligned}\hat{f}(\boldsymbol{k}) &= (f(0), f(1), \cdots, f(2^n - 1)) \\ &= (f(0), f(1), f(2) \cdots, f(2^n - 1))A_n = \boldsymbol{f}(\boldsymbol{x})A_n \end{aligned} \tag{4.4.8}$$

式(4.4.8)的逆变换为

$$\begin{aligned}\boldsymbol{f}(\boldsymbol{x}) &= (f(0), f(1), f(2) \cdots, f(2^n - 1)) \\ &= (f(0), f(1), f(2) \cdots, f(2^n - 1))A_n = \hat{\boldsymbol{f}}(\boldsymbol{k})A_n \end{aligned} \tag{4.4.9}$$

其中 A_n 由下式递归地来定义：

$$A_0 = [1]$$

$$A_n = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \otimes A_{n-1} = \begin{bmatrix} A_{n-1} & A_{n-1} \\ 0 & A_{n-1} \end{bmatrix}$$

$\otimes$表示矩阵的 Keronecker 积。易知，$A_n^2 = I$。

有一个快速算法能迭代地计算一个布尔函数 $f(x)$的 Reed-Muller 谱[40,80]。设 $\boldsymbol{f}^1(\boldsymbol{x})$ 和 $\boldsymbol{f}^2(\boldsymbol{x})$分别为向量 $\boldsymbol{f}$ 的前一半和后一半，那么

$$\hat{\boldsymbol{f}}(\boldsymbol{k}) = \boldsymbol{f}(\boldsymbol{x})A_n = (\boldsymbol{f}^1(\boldsymbol{x})A_{n-1}, (\boldsymbol{f}^1(\boldsymbol{x}) + \boldsymbol{f}^2(\boldsymbol{x}))A_{n-1})$$

直至迭代到 A_0 为止。详细讨论参阅文献[40,80]。

4.4.1.4 Walsh 谱表示

Walsh 谱是研究布尔函数的一个强有力的工具。这一点可由后面的一些讨论所证实。

定义 4.4.1 设 $x=(x_1, x_2, \cdots, x_n), w=(w_1, w_2, \cdots, w_n) \in F_2^n$，$x$ 和 w 的点积定义为 $w \cdot x = w_1x_1 + w_2x_2 + \cdots + w_nx_n \in F_2$。$n$ 个变元的布尔函数 $f(x)$的 Walsh 变换(又称 Walsh 谱)定义为

$$S_f(w) = 2^{-n}\sum_{x \in F_2^n} f(x)(-1)^{w \cdot x} \tag{4.4.10}$$

其逆变换为

$$f(x) = \sum_{w \in F_2^n} S_f(w)(-1)^{w \cdot x} \tag{4.4.11}$$

式(4.4.10)中将 $f(x)$视作实数，求和是指实数求和。式(4.4.11)称为布尔函数的 Walsh 谱表示，其中求和是指实数求和。注意式 (4.4.11)可由式(4.4.10)导出。$S_f(w)$通常称为 $f(x)$的线性 Walsh 谱。$f(x)$的另一种即所谓的循环 Walsh 谱定义为

$$S_{(f)}(w) = 2^{-n}\sum_{w \in F_2^n} (-1)^{f(x)}(-1)^{w \cdot x} \tag{4.4.12}$$

其逆变换为

$$f(x) = \frac{1}{2} - \frac{1}{2}\sum_{w \in F_2^n} S_{(f)}(w)(-1)^{w \cdot x} \tag{4.4.13}$$

由两种谱的定义并注意到 $(-1)^{f(x)} = 1 - 2f(x)$，直接可推出两种谱有如下关系：

$$S_{(f)}(w) = \begin{cases} -2S_f(w), w \neq 0 \\ 1 - 2S_f(w), w = 0 \end{cases} \tag{4.4.14}$$

设

$$f(x)=(f(0),f(1),\cdots,f(2^n-1))$$
$$S_f(w)=(S_f(0),S_f(1),\cdots,S_f(2^n-1))$$

则

$$S_f(w)=2^{-n}f(x)H_n$$

其中 H_n 由下式迭代地来定义：

$$H_0=[1]$$
$$H_n=\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\otimes H_{n-1}=\begin{bmatrix}H_{n-1} & H_{n-1}\\ H_{n-1} & -H_{n-1}\end{bmatrix}$$

$\otimes$表示矩阵的 Keronecker 积。因为 $H_n^2=2^nI_n$，所以 Walsh 逆变换为

$$f(x)=S_f(w)H_n$$

有计算布尔函数的 Walsh 谱的快速算法[41]。设 $f^1(x)$和 $f^2(x)$分别表示 $f(x)$的前一半和后一半，那么

$$S_f(w)=2^{-n}f(x)H_n=2^{-n}(f^1(x)H_{n-1}+f^2(x)H_{n-1},f^1(x)H_{n-1}-f^2(x)H_{n-1})$$

直至迭代到 H_0 为止。详细讨论参阅文献[41]。

由于关系式(4.4.14)，我们在用 Walsh 谱研究布尔函数时，可根据具体情况选用其中一种，在下面我们选用循环 Walsh 谱来研究布尔函数。

4.4.2 非线性度

非线性度是衡量布尔函数的非线性程度的一个重要指标，它刻划了一个布尔函数和线性布尔函数类之间的符合程度。其精确定义如下：

定义 4.4.2 设

$$f(x):F_2^n\to F_2,L_n=\{u\cdot x+v\mid u=(u_1,u_2,\cdots,u_n)\in F_2^n,v\in F_2\}$$

表示 F_2 上的所有线性布尔函数所成的集合(称为线性布尔函数类)。称非负整数

$$N_f=\min_{l(x)\in L_n}d_H(f(x),l(x))$$

为布尔函数 f 的非线性度。其中 $d_H(f(x),l(x))$表示 $f(x)$与 $l(x)$之间的汉明距离，即

$$d_H(f(x),l(x))=|\{x\in F_2^n\mid f(x)\neq l(x)\}|$$

对二元域，有 $d_H(f(x),l(x))=W_H(f+l)$。

有些文献用所谓的相关度来衡量布尔函数的非线性程度。f 的相关度定义为

$$C_f=\max_{l(x)\in L_n}|\{x\in F_2^n\mid f(x)=l(x)\}|$$

由非线性度和相关度的定义易知，$N_f+C_f=2^n$。该式表明，这两个指标本质上反映了同一件事情，即反映了 $f(x)$的非线性程度。

下面定理给出了布尔函数的非线性度的 Walsh 谱表示。

定理 4.4.1 设 $f(x):F_2^n\to F_2$ 的非线性度为 N_f，则

$$N_f=2^{n-1}\left(1-\max_{w\in F_2^n}|S_{(f)}(w)|\right)\qquad(4.4.15)$$

证明：由于$(-1)^vS_{(f)}(w)=\dfrac{1}{2^n}\sum_{x\in F_2^n}(-1)^{f(x)+w\cdot x+v}$

$$=\frac{1}{2^n}(|\{x\in F_2^n|f(x)=w\cdot x+v\}|)-(|\{x\in F_2^n|f(x)\neq w\cdot x+v\}|)$$

$$=\frac{1}{2^n}(2^n-2|\{x\in F_2^n|f(x)\neq w\cdot x+v\}|)$$

所以

$$d_H(f(x),w\cdot x+v)=|\{x\in F_2^n|f(x)\neq w\cdot x+v\}|$$
$$=2^{n-1}(1-(-1)^v S_{(f)}(w))$$

由定义 4.4.2 可知

$$N_f=\min_{l(x)\in L_n} d_H(f(x),l(x))=2^{n-1}(1-\max_{w\in F_2^n}|S_{(f)}(w)|)$$

由 $N_f+C_f=2^n$ 知

$$C_f=2^{n-1}(1+\max_{w\in F_2^n}|S_{(f)}(w)|) \tag{4.4.16}$$

式(4.4.15)或(4.4.16)表明，布尔函数 f 的非线性度和相关度由 f 的最大绝对谱值确定。同时，也说明了 f 的谱值实质上反映了该函数与线性函数之间的符合程度，亦即非线性程度或相关程度。

从密码学角度来讲，希望所选用的布尔函数 f 的非线性度越大越好(或相关度越小越好)。由式(4.4.15)可知，欲使 N_f 尽可能地大，$\max_{w\in F_2^n}|S_{(f)}(w)|$就必须尽可能地小。但

$$\sum_{w\in F_2^n} S_{(f)}^2(w)=1 \tag{4.4.17}$$

这是因为

$$\sum_{w\in F_2^n} S_{(f)}^2(w)=\sum_{x\in F_2^n} 2^{-n}\sum_{w\in F_2^n}(-1)^{f(x)+w\cdot x}2^{-n}\sum_{y\in F_2^n}(-1)^{f(y)+w\cdot y}$$

$$=2^{-2n}\sum_{x,y\in F_2^n}(-1)^{f(x)+f(y)}\sum_{w\in F_2^n}(-1)^{w\cdot(x+y)}$$

$$=2^{-2n}\sum_{x\in F_2^n}1\sum_{w\in F_2^n}1=2^{-2n}\times 2^n\times 2^n=1$$

所以，$\max_{w\in F_2^n}|S_{(f)}(w)|\geqslant 2^{-\frac{n}{2}}$，因此 $N_f\leqslant 2^{n-1}(1-2^{-\frac{n}{2}})$。当$\max_{w\in F_2^n}|S_{(f)}(w)|$取最小值 $2^{-\frac{n}{2}}$时，N_f 达到最大值 $2^{n-1}(1-2^{-\frac{n}{2}})$，此时对一切 $w\in F_2^n$，都有 $|S_{(f)}(w)|=2^{-\frac{n}{2}}$，人们将这类函数称为 Bent 函数。Bent 函数与线性函数类中的每个函数的符合率都一样。对 Bent 函数 f，$N_f=2^{n-1}(1-2^{-\frac{n}{2}})=2^{n-1}-2^{\frac{n}{2}-1}$，因为 N_f 为正整数，所以 $\frac{n}{2}$ 必为正整数，说明 n 为偶数。故 Bent 函数存在的必要条件是 n 为偶数。这里介绍几种构造 Bent 函数的方法。

方法 1 设 $n=2m$，g 是任意一个 m 元布尔函数，令

$$f(x_1,x_2,\cdots,x_n)=g(x_1,\cdots,x_m)+x_1x_{m+1}+\cdots+x_mx_n$$

则 f 是一个 n 元 Bent 函数。通过直接计算 f 的谱值 $S_{(f)}(w)$来说明 f 是一个 n 元 Bent 函数。

方法 2 设

$$x=(x_1,x_2,\cdots,x_n),a(x),b(x),c(x),a(x)+b(x)+c(x)$$

都是 Bent 函数，令

$$f(x,x_{n+1},x_{n+2})=a(x)b(x)+a(x)c(x)+b(x)c(x)+(a(x)+b(x))x_{n+1}$$
$$+(a(x)+c(x))x_{n+2}+x_{n+1}x_{n+2}$$

则 f 为一个 Bent 函数。

通过直接计算 f 的谱值 $S_{(f)}(w)$可说明 f 是一个 Bent 函数。

方法 3 设 $g:F_2^n\to F_2$ 是任意一个布尔函数，$\pi:F_2^n\to F_2^n$ 是任意一个置换，令

$$f:F_2^{2n}=F_2^n\times F_2^n\to F_2 \qquad f(x',x'')=\pi(x')\cdot x''+g(x')$$

则 f 为一个 Bent 函数。也可以通过直接计算 f 的谱值 $S_{(f)}(w)$可说明 f 是一个 Bent 函数。作为一个例子，我们来详细推导如下：

$$\begin{aligned}S_{(f)}(w',w'')&=2^{-2n}\sum_{(x',x'')\in F_2^{2n}}(-1)^{f+w'\cdot x'+w''\cdot x''}\\&=2^{-2n}\sum_{x'\in F_2^n}(-1)^{g(x')+w'\cdot x'}\sum_{x''\in F_2^n}(-1)^{\pi(x')\cdot x''+w''\cdot x''}\\&=2^{-2n}\sum_{x'\in F_2^n,\pi(x')=w''}(-1)^{g(x')+w'\cdot x'}\cdot 2^n\\&=2^{-n}(-1)^{g(\pi^{-1}(w''))+w'\cdot\pi^{-1}(w'')}\end{aligned}$$

则对一切 $w=(w',w'')\in F_2^{2n}$，有 $|S_{(f)}(w)|=2^{-n}=2^{-\frac{2n}{2}}$，故 $f(x)$是一个 Bent 函数。

文献[16,156]中给出了 Bent 函数和组合学中的差集之间的关系，这样我们就可以通过差集来构造和研究 Bent 函数。关于 Bent 函数的构造和分析可进一步参阅文献[42,43,44,45,157]。

从密码学角度来看，Bent 函数也存在着一些缺陷，诸如它不是平衡的，它的非线性次数不超过 $n/2$[46]，限制 n 为偶数。这就要求对 Bent 函数进行改进，因此，构造非线性度高、非线性次数大的平衡布尔函数和构造非线性度高的奇数个变元的布尔函数等问题就成了人们所关注的问题，详细讨论请参阅文献[35,47,48,49,50]。另一个值得注意的问题是布尔函数的非线性度的研究与编码学中的 1 阶 Reed-Muller 码的覆盖半径的研究等价，这样我们可以利用已知的关于 1 阶 Reed-Muller 码的覆盖半径的研究结果[51,52]构造出一批具有较高非线性度的布尔函数。文献[35,53,54]中给出了布尔函数 m 阶 Walsh 谱的定义，讨论了布尔函数的 m 次逼近以及 m 次 Bent 函数等问题，限于篇幅，这里就不在阐述。本小节最后我们给出布尔函数的非线性度和其谱值为零的个数之间的关系。

设 $f:F_2^n\to F_2$ 的非线性度为 N_f，记

$$N_{s_{(f)}}=|\{w\in F_2^n|S_{(f)}(w)=0\}|$$

则

$$N_f/2^{n-1}+1/\sqrt{2^n-N_{s_{(f)}}}\leqslant 1 \tag{4.4.18}$$

式(4.4.18)可由式(4.4.17)和(4.4.15)直接推出。

式(4.4.18)表明，N_f 和 $N_{s_{(f)}}$之间存在一种制约关系。

4.4.3 线性结构和退化性

4.4.2 节中，我们讨论了一个布尔函数和密码学上的弱的线性布尔函数类之间的“相似”程度，即把线性布尔函数类作为参照系，考察一个给定的函数与它的“远近”程度。我们

用所谓的非线性度或相关度来衡量这种"相似"或"远近"程度。密码学上弱的另一类函数是具有线性结构的函数类，本节我们不打算讨论一个布尔函数和这类函数之间的相关或符合程度，感兴趣的读者请参阅文献[55]，只讨论这类函数的结构并说明它与退化性之间的关系。

定义 4.4.3 设 $f(x):F_2^n\to F_2$，$\alpha\in F_2^n$，如果对一切 $x\in F_2^n$，都有 $f(x+\alpha)-f(x)=$常值$(=f(\alpha)-f(0))$，则称 α 为 $f(x)$的一个线性结构。

令 $f(x)$的全体线性结构之集为 U_f，直接可验证 U_f 为 F_2^n 的一个线性子空间。再令

$$U_f^{(0)}=\{\alpha\in F_2^n\mid f(x+\alpha)-f(x)=0,\text{对一切 } x\in F_2^n\}$$

$$U_f^{(1)}=\{\alpha\in F_2^n\mid f(x+\alpha)-f(x)=1,\text{对一切 } x\in F_2^n\}$$

显然 $U_f=U_f^{(0)}\cup U_f^{(1)}$。直接可验证 $U_f^{(0)}$ 为 U_f 的一个线性子空间，当然，也是 F_2^n 的一个线性子空间。

设 $\alpha\in U_f^{(1)}$ 则易知

$$U_f^{(1)}=\alpha+U_f^{(0)}=\{\alpha+\beta\mid\beta\in U_f^{(0)}\}$$

从而 $U_f=U_f^{(0)}\cup(\alpha+U_f^{(0)})$。这表明，我们只要研究清楚 $U_f^{(0)}$ 的结构，U_f 的结构也就清楚了。下面定理给出了布尔函数的线性结构的特征。

定理 4.4.2 设 $f(x):F_2^n\to F_2$ 的线性结构之集为 $U_f=U_f^{(0)}\cup U_f^{(1)}$，则 $\alpha\in U_f^{(i)}(i=0$ 或 $1)$，当且仅当对任意的 $w\in F_2^n:w\cdot\alpha=\bar{i}=i+1$，即 $w\cdot\alpha\neq i$，有 $S_{(f)}(w)=0$。

证明：设 $\alpha\in U_f^{(i)}(i=0$ 或 $1)$，由定义知

$$\begin{aligned}S_{(f)}(w)&=2^{-n}\sum_{x\in F_2^n}(-1)^{f(x)+w\cdot x}\\&=2^{-n}\sum_{x\in F_2^n}(-1)^{f(x+\alpha)+w\cdot(x+\alpha)}=2^{-n}\sum_{x\in F_2^n}(-1)^{f(x)+i+w\cdot(x+\alpha)}\\&=(-1)^{w\cdot\alpha+i}2^{-n}\sum_{x\in F_2^n}(-1)^{f(x)+w\cdot x}=(-1)^{w\cdot\alpha+i}S_{(f)}(w)\end{aligned}$$

若 $w\cdot\alpha+i\neq0$，则 $w\cdot\alpha\neq i$，即 $w\cdot\alpha=\bar{i}=i+1$，由上式知，$S_{(f)}(w)=0$。反之，对 $\alpha\in F_2^n$，

$$S_{(f(x+\alpha))}(w)=2^{-n}\sum_{x\in F_2^n}(-1)^{f(x+\alpha)+w\cdot x}=2^{-n}\sum_{x\in F_2^n}(-1)^{f(x)+w\cdot(x+\alpha)}=(-1)^{w\cdot\alpha}S_{(f)}(w)$$

$$S_{(f+i)}(w)=2^{-n}\sum_{x\in F_2^n}(-1)^{f(x)+i+w\cdot x}=(-1)^iS_{(f)}(w)$$

当 $w\cdot\alpha=i$ 时，由上式知，$S_{(f(x+\alpha))}(w)=S_{(f+i)}(w)$。

当 $w\cdot\alpha\neq i$，即 $w\cdot\alpha=i+1$，由假设条件可知，$S_{(f)}(w)=0$，从而由上式可知 $S_{(f(x+\alpha))}(w)=0$，$S_{(f+i)}(w)=0$，即 $S_{(f(x+\alpha))}(w)=S_{(f+i)}(w)$。说明对一切 $w\in F_2^n$，有 $S_{(f(x+\alpha))}(w)=S_{(f+i)}(w)$，由式(4.4.13)可知，对一切 $x\in F_2^n$，有 $f(x+\alpha)=f+i$，故 $\alpha\in U_f^{(i)}$。

若 $U_f^{(1)}\neq\varnothing$，即有 $\alpha\in U_f^{(1)}$，由定理 4.4.2 可知，对一切 $w\in F_2^n:w\cdot\alpha=0$，都有 $S_{(f)}(w)=0$。特别地，对 $w=0$，有 $S_{(f)}(0)=0$。又由 Walsh 谱 $S_{(f)}(w)$的定义易知，f 是平衡的，当且仅当 $S_{(f)}(0)=0$。故 $U_f^{(1)}\neq\varnothing$的必要条件是 f 为平衡的。

下面我们来给出 $U_f^{(0)}$ 的结构。

定理 4.4.3 $U_f^{(0)}=\langle\{w\in F_2^n\mid S_{(f)}(w)\neq0\}\rangle^{\perp}$，其中$\langle\{w\in F_2^n\mid S_{(f)}(w)\neq0\}\rangle$表示由集

合$\{w\in F_2^n|S_{(f)}(w)\neq 0\}$生成的线性子空间，即

$$\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$$
$$=\left\{w\in F_2^n|w=\sum_j w_j, S_{(f)}(w_j)\neq 0\right\}, \langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle^{\perp}$$

表示线性子空间$\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$的正交空间，即

$\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle^{\perp}=\{\alpha\in F_2^n|\beta\cdot\alpha=0$，对一切$\beta\in\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$。

证明：$\alpha\in U_f^{(0)}$，由定理 4.4.2 可知，对一切 $w: w\cdot\alpha=1$，有 $S_{(f)}(w)=0$，于是对任意的 $w\in\{w\in F_2^n|S_{(f)}(w)\neq 0\}$，都有 $w\cdot\alpha=0$。而$\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$是由$\{w\in F_2^n|S_{(f)}(w)\neq 0\}$生成的线性子空间，所以对一切$w\in\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$，都有$w\cdot\alpha=0$，故$\alpha\in\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle^{\perp}$。这表明$U_f^{(0)}\subseteq\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle^{\perp}$。

反之，设$\alpha\in\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle^{\perp}$，则对一切$w\in\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$，都有$w\cdot\alpha=0$，从而$\langle\{\alpha\}\rangle^{\perp}\supseteq\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle$。因此，对任意的$w\notin\langle\{\alpha\}\rangle^{\perp}$，即$w\cdot\alpha\neq 0$，

$$w\notin\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle\supset\{w\in F_2^n|S_{(f)}(w)\neq 0\}$$

所以$S_{(f)}(w)=0$，由定理 4.4.2 可知，$\alpha\in U_f^{(0)}$。综上所述

$$U_f^{(0)}=\langle\{w\in F_2^n|S_{(f)}(w)\neq 0\}\rangle^{\perp}$$

现在我们来讨论布尔函数的线性结构和非线性度之间的关系。

定理 4.4.4 设$f(x): F_2^n\to F_2$的线性结构$U_f=U_f^{(0)}\cup U_f^{(1)}$，非线性度为$N_f$，记$\dim U_f=k$，则

$$N_f/2^{n-1}+1/\sqrt{2^{n-k}}\leqslant 1 \tag{4.4.19}$$

证明：由式(4.4.18)可知，我们要证明式(4.4.19)，只需证明$2^n-N_{S_{(f)}}\leqslant 2^{n-k}$。因为$U_f=U_f^{(0)}\cup U_f^{(1)}$是一个$k$维线性子空间，若$U_f^{(1)}\neq\varnothing$，则$U_f^{(0)}$是一个$k-1$维线性子空间。设$\beta_1,\beta_2,\cdots,\beta_{k-1}\in U_f^{(0)}$且$\beta_1,\beta_2,\cdots,\beta_{k-1}$在$F_2$上线性无关，取$\beta_k\in U_f^{(1)}$，易验证$\beta_1,\beta_2,\cdots,\beta_{k-1},\beta_k$在$F_2$上线性无关。又

$$f(x)=f(x+\beta_1)=\cdots=f(x+\beta_{k-1})=1+f(x+\beta_k)$$

则

$$S_{(f)}(w)=(-1)^{\beta_1\cdot w}S_{(f)}(w)=\cdots=(-1)^{\beta_{k-1}\cdot w}S_{(f)}(w)=(-1)^{1+\beta_k\cdot w}S_{(f)}(w)$$

因此，如果$S_{(f)}(w)\neq 0$，那么$\beta_1\cdot w=\cdots=\beta_{k-1}\cdot w=1+w\cdot\beta_k=0$。由$\beta_1,\beta_2,\cdots,\beta_{k-1},\beta_k$的线性无关性可知，这个方程组有$2^{n-k}$个解。故

$$|\{w\in F_2^n: S_{(f)}(w)\neq 0\}|=2^n-N_{S_{(f)}}\leqslant 2^{n-k}$$

若$U_f^{(1)}=\varnothing$，此时，$U_f=U_f^{(0)}$，用类似于上述的方法可证明$2^n-N_{S_{(f)}}\leqslant 2^{n-k}$。

式(4.4.19)表明，布尔函数的非线性度和线性结构之间存在着一种制约关系。

在逻辑电路和密码函数的设计中，经常考虑逻辑函数与变元之间的代数无关性。所谓一个布尔函数$f(x): F_2^n\to F_2$与$x_{i_1},x_{i_2},\cdots,x_{i_r}$代数无关，是指对一切$\alpha_1,\alpha_2,\cdots,\alpha_r\in F_2$，有

$$f(x_1,x_2,\cdots,x_n)=f(x_1,\cdots,x_{i_1}+\alpha_1,\cdots,x_{i_r}+\alpha_r,\cdots,x_n)$$

显然，如果$f(x)$与变元$x_{i_1},x_{i_2},\cdots,x_{i_r}$代数无关，则$V_{i_1i_2\cdots i_r}=\{\alpha=(0,\cdots,\alpha_1,\cdots,\alpha_r,\cdots)|\alpha$的第$i_1,i_2,\cdots,i_r$个分量在$F_2$上任意取值，其它分量全为$0\}\subseteq U_f^{(0)}$，可见布尔函数与其变元之间的代数无关性的研究可归纳为布尔函数的线性结构的研究。

逻辑函数的退化性可给密码分析者带来益处，因此，需对逻辑函数的退化性进行研

究。布尔函数的退化性的精确定义如下。

定义 4.4.4 设 $f(x):F_2^n\to F_2$，如果存在 F_2 上的一个 $k\times n(k<n)$ 阶矩阵 D 和 F_2^k 到 F_2 上的一个函数 $g(y)$，使得 $f(x)=g(Dx)=g(y)$，则称 $f(x)$是退化的，其中 $y=Dx$。

注意，在定义 4.4.4 中，$y=Dx$ 实际上是一个列向量，为了叙述方便起见，我们将列向量和行向量一样看待，以下同。

下面我们来说明布尔函数的线性结构和退化性之间的关系。

定理 4.4.5 设 $f(x):F_2^n\to F_2$，f 的线性结构之集为 $U_f=U_f^{(0)}\cup U_f^{(1)}$，则 $\dim U_f^{(0)}=k$，当且仅当 $f(x)$能且至多能退化为 $n-k$ 个变元的函数。其中 $\dim U_f^{(0)}$ 表示 $U_f^{(0)}$ 的维数。

证明：设 $f(x)$能且至多能退化为 $n-k$ 个变元的函数，则由定义 4.4.4 可知，存在$(n-k)\times n$ 阶矩阵 D 和函数 $g(y):F_2^{n-k}\to F_2$，使得对一切 $x\in F_2^n$，$f(x)=g(Dx)=g(y)$，秩 $D=n-k$。

令 $\ker D=\{x\in F_2^n\mid Dx=0\}$，显然 $\ker D$ 是 F_2^n 的一个线性子空间，且 $\dim(\ker D)=k$。下面证明 $U_f^{(0)}=\ker D$。显然，$\ker D\subseteq U_f^{(0)}$。为了证明 $U_f^{(0)}\subseteq\ker D$，先证明结论：若 $\dim U_f^{(0)}=r$，则 f 能退化为 $n-r$ 个变元的函数。由代数知识可知，此时 F_2^n 可分解为

$$F_2^n=\bigcup_{\beta_i\in S}(\beta_i+U_f^{(0)})$$

其中 S 为代表元集，$|S|=2^{n-r}$。直接可验证 $f(x)$在 $U_f^{(0)}$ 的每个陪集上取值为常值。定义映射 $\varphi:S\to F_2^{n-r}$，$\varphi(\beta)=D\beta$，其中 D 是由 $U_f^{(0)}$ 的正交空间$(U_f^{(0)})^{\perp}$的一组基所构成的$(n-r)\times n$ 阶矩阵。设 $\beta_1,\beta_2\in S$，若 $\varphi(\beta_1)=\varphi(\beta_2)$，则 $D(\beta_1-\beta_2)=0$，即 $\beta_1-\beta_2\in((U_f^{(0)})^{\perp})^{\perp}=U_f^{(0)}$，说明 $\beta_1=\beta_2$，从而 φ 是单射。又 $|S|=2^{n-r}=|F_2^{n-r}|$，故 φ 是双射。定义函数 $g(y):F_2^{n-r}\to F_2$，$g(y)=f(\varphi^{-1}(y))$，则对任意的 $\beta\in S$，有

$$g(D\beta)=f(\varphi^{-1}(D\beta))=f(\beta)$$

对任意的 $x\in F_2^n$，存在 $\beta\in S$ 和 $d\in U_f^{(0)}$ 使 $x=\beta+d$，因此

$$Dx=D(\beta+d)=D\beta$$

又 $f(x)$在 $U_f^{(0)}$ 的每个陪集上取值为常值，所以

$$f(x)=f(\beta)=g(D\beta)=g(Dx)$$

即 f 能退化为 $n-r$ 个变元的函数。下面来证明 $U_f^{(0)}\subseteq\ker D$。设 $\alpha\in U_f^{(0)}$，则对任意的 $x\in F_2^n$，有 $f(x+\alpha)=f(x)$，又

$$f(x+\alpha)=g(D(x+\alpha))=g(y+D\alpha)\qquad f(x)=g(y)$$

所以 $g(y+D\alpha)=g(y)$，由于 $g(y)$不能再退化，所以由上面证明的结论可知，$D\alpha=0$，即 $\alpha\in\ker D$。综上所述，$U_f^{(0)}=\ker D$，故 $\dim(\ker D)=\dim U_f^{(0)}=k$。

反之，设 $\dim U_f^{(0)}=k$，由上面的结论可知，f 能退化为 $n-k$ 个变元的函数，再由充分性可知，f 能且至多能退化为 $n-k$ 个变元的函数。

定理 4.4.5 表明，布尔函数的退化性的研究也可以归结为布尔函数的线性结构的研究。

文献[35]中推广了布尔函数的线性结构的概念并对其作了一些研究。文献[56]中讨论了具有线性结构的布尔函数的计数问题。布尔函数的线性结构与其它一些概念之间的关系可参阅文献[57]。

4.4.4 严格雪崩准则和扩散准则

文献[58]中在研究 S-盒的设计时，将“完全性”和“雪崩效应”这两个概念进行组合定义了一个新的概念——严格雪崩准则(strict avalanche criterion)，简记为 SAC。文献[59，60，61，62]对高阶严格雪崩准则进行了研究。在这里我们仅讨论文献[58]中定义的布尔函数的严格雪崩准则(SAC)，对高阶的情形感兴趣的读者请参阅有关文献。

定义 4.4.5 设 $f(x):F_2^n\to F_2$，如果对任何 $e_i=(0,\cdots,0,1,0,\cdots,0)\in F_2^n$，$1\leqslant i\leqslant n$，都有 $f(x)+f(x+e_i)$是一个平衡布尔函数，则称 $f(x)$满足严格雪崩准则，其中 e_i 的第 i 个分量为 1，其余分量均为 0。

设 $f(x):F_2^n\to F_2$，我们称 $C_f(s)=\sum_{x\in F_2^n}(-1)^{f(x+s)+f(x)}$为 f 的自相关函数。

由自相关函数的定义可知，$f(x)$满足严格雪崩准则，当且仅当对任意的 $e_i\in F_2^n$，$1\leqslant i\leqslant n$，有 $C_f(e_i)=0$。

由定义直接可证明 $C_f(s)$和 $f(x)$的 Walsh 谱之间有如下关系：

$$S_{C_f}(w)=2^{-n}\sum_{s\in F_2^n}C_f(s)(-1)^{s\cdot w}=2^nS_{(f)}^2(w) \tag{4.4.20}$$

由式(4.4.20)可推出

$$C_f(s)=2^n\sum_{w\in F_2^n}S_{(f)}^2(w)(-1)^{w\cdot s} \tag{4.4.21}$$

由式(4.4.21)可知，$f(x)$满足严格雪崩准则，当且仅当对任意的 $e_i\in F_2^n$，$1\leqslant i\leqslant n$，有

$$\sum_{w\in F_2^n}S_{(f)}^2(w)(-1)^{w\cdot e_i}=0$$

令

$$N_{C_f}=|\{s\in F_2^n|C_f(s)=0\}|$$

文献[63]猜测 N_{C_f}和 $N_{S_{(f)}}$有如下关系：

$$(2^n-N_{C_f})(2^n-N_{S_{(f)}})\geqslant 2^n$$

文献[64]证明了这个猜测是正确的，并把等号成立的函数定义成部分 Bent 函数且对这种函数的结构进行了刻划。

文献[65]将文献[59]引入的“50%-依赖性”的概念和“完全非线性性”的概念进行了一般化，引入了高次扩散准则(propagation criteria)的概念。

定义 4.4.6 设 $f(x):F_2^n\to F_2$，如果 $\alpha\in F_2^n$，使得 $f(x+\alpha)+f(x)$是一个平衡布尔函数，则称 $f(x)$关于 α 满足扩散准则。如果对所有的向量 $\alpha\in F_2^n:1\leqslant W_H(\alpha)\leqslant k$，$f(x)$关于 α 满足扩散准则，则称 $f(x)$满足 k 次扩散准则。

由定义可知，1 次扩散准则的概念和严格雪崩准则的概念是一致的。

由自相关函数的定义可知，$f(x)$关于 $\alpha\in F_2^n\backslash\{0\}$满足扩散准则，当且仅当 $C_f(\alpha)=0$。$f(x)$满足 k 次扩散准则，当且仅当对任意的 $\alpha\in F_2^n:1\leqslant W_H(\alpha)\leqslant k$，有 $C_f(\alpha)=0$。

由式(4.4.21)可推出，$f(x)$关于 $\alpha\in F_2^n\backslash\{0\}$满足扩散准则，当且仅当

$$\sum_{w\in F_2^n}S_{(f)}^2(w)(-1)^{w\cdot\alpha}=0$$

$f(x)$满足 k 次扩散准则，当且仅当对所有的 $a\in F_2^n$：$1\leqslant W_H(a)\leqslant k$，有 $\sum_{w\in F_2^n} S^2_{(f)}(w)(-1)^{w\cdot a}=0$。

当 $f(x)$关于 $a\in F_2^n\backslash\{0\}$满足扩散准则时，不妨设 a 的非零分量为 $i_1,i_2,\cdots,i_s$，$f(x)$关于它的变元 $x_{i_1},x_{i_2}\cdots,x_{i_s}$是 50%-依赖的；当 $f(x)$满足 n 次扩散准则时，由式(4.4.20)可知，$f(x)$是 Bent 函数，亦称 $f(x)$是完全非线性的。由此可见，扩散准则的概念是“50%-依赖性”的概念和“完全非线性性”的概念的一般化。

关于扩散准则的进一步研究和别的定义，可参阅文献[63,66]。

4.4.5 相关免疫性

关于前面讨论的逻辑函数的线性结构、退化性、与变元的无关性等性质的研究属于逻辑函数的代数性质。本小节我们来讨论逻辑函数与其变元的统计特性，即与变元的统计无关性和相关免疫性等性质。研究逻辑函数的统计无关性的意义已在文献[47,67]等进行了论述。

定义 4.4.7 设 $f(x):F_2^n\to F_2$，$x_1,x_2,\cdots,x_n$ 是 F_2 上的独立的、均匀分布的随机变量，如果对任意的$(a_1,a_2,\cdots,a_m)\in F_2^m(m\leqslant n)$及 $a\in F_2$，都有 $p(f=a,x_{i_1}=a_1,x_{i_2}=a_2,\cdots,x_{i_m}=a_m)=\frac{1}{2^m}p(f=a)$，则称 f 与变元 $x_{i_1},x_{i_2}\cdots,x_{i_m}$统计无关。如果 f 与 $x_1,x_2,\cdots,x_n$ 中的任意 m 个变元都统计无关，则称 f 是 m 阶相关免疫的。

下面定理给出了 f 与其变元 $x_{i_1},x_{i_2},\cdots,x_{i_m}$统计无关的一些等价条件。

定理 4.4.6 设 $f(x)$如定义 4.4.7 所述，则下列三个条件等价：

(1)$f(x)$与变元 $x_{i_1},x_{i_2},\cdots,x_{i_m}$统计无关；

(2)对任意的 $w=(0,\cdots,w_{i_1},\cdots,w_{i_m},\cdots,0)\in F_2^n$：$1\leqslant W_H(w)\leqslant m$，$f(x)$与 $w\cdot x$ 统计无关；

(3)对任意的 $w=(0,\cdots,w_{i_1},\cdots,w_{i_m},\cdots,0)\in F_2^n$：$1\leqslant W_H(w)\leqslant m$，$f(x)+w\cdot x$ 是平衡的。

证明：(1)⇒(2)，显然成立。

(2)⇒(3)，设对任意的 $w=(0,\cdots,w_{i_1},\cdots,w_{i_m},\cdots,0)\in F_2^n$：$1\leqslant W_H(w)\leqslant m$，$f(x)$与 $w\cdot x$统计无关，则对任意的 $i\in F_2$

$$\begin{aligned}p(f(x)+w\cdot x=i)&=\sum_{a\in F_2}p(f(x)=a,w\cdot x=i-a)\\&=\sum_{a\in F_2}p(f(x)=a)p(w\cdot x=i-a)\\&=\frac{1}{2}\sum_{a\in F_2}p(f(x)=a)=\frac{1}{2}\end{aligned}$$

从而

$$W_H(f+w\cdot x)=|\{x\in F_2^n|f(x)+w\cdot x=1\}|=2^n\times\frac{1}{2}=2^{n-1}$$

故 $f(x)+w\cdot x$ 是平衡的。

(3)⇒(1) 设对任意的 $w=(0,\cdots,w_{i_1},\cdots,w_{i_m},\cdots,0)\in F_2^n$：$1\leqslant W_H(w)\leqslant m$，$f(x)+w$

$\cdot x$ 是平衡的。对任意的 $a,a_1,a_2,\cdots,a_m\in F_2$，记 $A=(a,a_1,a_2,\cdots,a_m)$，$F(x)=(f(x),x_{i_1},x_{i_2},\cdots,x_{i_m})$，$N_A=|\{x\in F_2^n|F(x)=A\}|$，$N_a=|\{x\in F_2^n|f(x)=a\}|$。因为

$$\sum_{x\in F_2^n}\sum_{y\in F_2^{m+1}}(-1)^{A\cdot y+F(x)\cdot y}=\sum_{y\in F_2^{m+1}}(-1)^{A\cdot y}\sum_{x\in F_2^n}(-1)^{F(x)\cdot y}$$

$$=2^n+\sum_{y\in F_2^{m+1}\setminus\{0\}}(-1)^{A\cdot y}\sum_{x\in F_2^n}(-1)^{F(x)\cdot y}$$

利用假设条件可知，当 $y\neq(0,0,\cdots,0),(1,0,\cdots,0)$ 时

$$\sum_{x\in F_2^n}(-1)^{F(x)\cdot y}=0$$

所以

$$\sum_{x\in F_2^n}\sum_{y\in F_2^{m+1}}(-1)^{A\cdot y+F(x)\cdot y}=2^n+(-1)^a\sum_{x\in F_2^n}(-1)^{f(x)}$$

$$=2^n+\sum_{x\in F_2^n}(-1)^{f(x)+a}=2N_a$$

又

$$\sum_{x\in F_2^n}\sum_{y\in F_2^{m+1}}(-1)^{A\cdot y+F(x)\cdot y}=\sum_{x\in F_2^n}\sum_{y\in F_2^{m+1}}(-1)^{(A+F(x))\cdot y}=N_A\times 2^{m+1}$$

故由上述两式可得 $N_A\times 2^m=N_a$，即

$$p(f=a,x_{i_1}=a_1,x_{i_2}=a_2,\cdots,x_{i_m}=a_m)$$

$$=\frac{1}{2^m}p(f=a)=p(f=a)p(x_{i_1}=a_1)\cdots p(x_{i_m}=a_m)$$

由 A 的任意性可知，$f(x)$ 与变元 $x_{i_1},x_{i_2},\cdots,x_{i_m}$ 统计无关。

由定理 4.4.6 立即可推出 $f(x)$ 与变元 $x_{i_1},x_{i_2},\cdots,x_{i_m}$ 统计无关的谱特征。

定理 4.4.7 设 $f(x)$ 如定义 4.4.7 所述，则 $f(x)$ 与变元 $x_{i_1},x_{i_2},\cdots,x_{i_m}$ 统计无关，当且仅当对任意的 $w=(0,\cdots,w_{i_1},\cdots,w_{i_m},\cdots,0)\in F_2^n:1\leqslant W_H(w)\leqslant m,S_{(f)}(w)=0$。

证明：由定理 4.4.6 可知，$f(x)$ 与变元 $x_{i_1},x_{i_2},\cdots,x_{i_m}$ 统计无关，当且仅当对任意的 $w=(0,\cdots,w_{i_1},\cdots,w_{i_m},\cdots,0)\in F_2^n:1\leqslant W_H(w)\leqslant m,f(x)+w\cdot x$ 是平衡的，而 $f(x)+w\cdot x$ 是平衡的，当且仅当 $S_{(f+w\cdot x)}(0)=S_{(f)}(w)=0$。从而定理 4.4.7 得证。

由定理 4.4.6 和定理 4.4.7 立即可得到如下两个定理。

定理 4.4.8 设 $f(x)$ 如定义 4.4.7 所述，则下列三个条件等价：

(1) $f(x)$ 是 m 阶相关免疫的；

(2) 对任意的 $w\in F_2^n:1\leqslant W_H(w)\leqslant m,f(x)$ 与 $w\cdot x$ 统计无关；

(3) 对任意的 $w\in F_2^n:1\leqslant W_H(w)\leqslant m,f(x)+w\cdot x$ 是平衡的。

定理 4.4.9 设 $f(x)$ 如定义 4.4.7 所述，则 $f(x)$ 是 m 阶相关免疫的，当且仅当对任意的 $w=\in F_2^n:1\leqslant W_H(w)\leqslant m,S_{(f)}(w)=0$。

由定理 4.4.9 和式(4.4.14)立即可推出如下著名的 Xiao-Massey 定理：

定理 4.4.10[68] 设 $f(x)$ 如定义 4.4.7 所述，则 $f(x)$ 是 m 阶相关免疫的，当且仅当对任意的 $w\in F_2^n:1\leqslant W_H(w)\leqslant m,S_f(w)=0$。

记 $b_i=|\{x\in F_2^n|f(x)=i\}|,0\leqslant i\leqslant 1$，取 $y=(y_{i_1},y_{i_2},\cdots,y_{i_m})$，记

$$a_i(y) = |\{x \in F_2^n | f(x_1,\cdots,y_{i_1},\cdots,y_{i_m},\cdots,x_n) = i\}|$$

若 $f(x)$是 m 阶相关免疫的，则对任意固定的 $y\in F_2^m$，有 $a_i(y)=b_i/2^m$，$0\leqslant i\leqslant 1$。再记由 $W_i=\{x\in F_2^n|f(x)=i\}$中的元素作为行所构成的矩阵为 B_i，B_i 是一个 $b_i\times n$ 阶矩阵。

定义 4.4.8 设 A 是 F_2 上的一个 $M\times n$ 阶矩阵，如果对任意给定的 m 列，每一个行向量 $\alpha\in F_2^m$ 恰好重复 $M/2^m$ 次，则称 A 为正交矩阵，记为$(M,n,2,m)$。

显然，若 A 为$(M,n,2,m)$正交矩阵，则对任意的 $r\leqslant m$，A 亦为$(M,n,2,r)$正交矩阵。

由上面的讨论可知，B_i 是一个$(b_i,n,2,m)$正交矩阵。反之，若 B_i 是一个$(b_i,n,2,m)$正交矩阵，则对任意的 $y\in F_2^m$，有 $a_i(y)=b_i/2^m$。因此有如下定理：

定理 4.4.11 设 $f(x)$如定义 4.4.7 所述，则 $f(x)$是 m 阶相关免疫的，当且仅当 B_i $(0\leqslant i\leqslant 1)$是$(b_i,n,2,m)$正交矩阵。

由定理 4.4.11 可知，$f(x)$是 m 阶相关免疫的必要条件是 b_i 是 2^m 的倍数，特别地，$W_H(f)$是 2^m 的倍数。事实上，定理 4.4.11 中只要 B_0 和 B_1 中之一是一个$(b_i,n,2,m)$正交矩阵即可推得 $f(x)$是 m 阶相关免疫的。

定理 4.4.11 给出了相关免疫阶和组合学中的正交矩阵之间的关系，从而给出了相关免疫阶和组合设计之间的关系。该定理在相关免疫函数的构造和计数的研究中起着重要作用。关于相关免疫阶和编码中的对偶距离之间的关系参见文献[69]。

已有大量文献对相关免疫函数的构造与计数问题进行了讨论，这里仅仅介绍 Siegenthaler[70]给出的一种迭代构造方法。对相关免疫函数的构造和计数的研究感兴趣的读者可进一步参阅文献[48,71,72]等。

方法：设 f_1 和 f_2 是两个 n 个变元的 m 阶相关免疫函数，令

$$f(x_1,x_2,\cdots,x_{n+1}) = x_{n+1}f_1(x_1,\cdots,x_n) + \overline{x_{n+1}}f_2(x_1,\cdots,x_n)$$

则 f 是一个 $n+1$ 个变元的 m 阶相关免疫函数

$$\partial^0 f = \max\{\partial^0 f_1, \partial^0 f_2\} + 1$$

易知 $f:F_2^n\to F_2$ 是 $n-1$ 阶相关免疫函数的充要条件是 $f(x)=x_1+x_2+\cdots+x_n+c, c\in F_2$。

下面我们来讨论布尔函数的相关免疫阶和非线性度以及非线性次数之间的关系。

由定理 4.4.9 和式(4.4.18)易知，$f(x)$的相关免疫阶 m 和其非线性度 N_f 之间有如下关系：

$$N_f/2^{n-1} + 1 \Big/ \sqrt{2^n - \sum_{i=1}^{m} C_n^i} \leqslant 1$$

此式已表明 m 与 N_f 之间存在某种制约关系。不过我们在这里给出一个比较紧的关系。

定理 4.4.12 设 $f(x)$如定义 4.4.7 所述，$f(x)$是 m 阶相关免疫的，但不是 $m+1$ 阶相关免疫的，$f(x)$的非线性度为 N_f，f 的汉明重量为 $W_H(f)=2^m\cdot k_0$(k_0 为一非负整数)，则存在正整数 a，使得

$$N_f/2^{n-1} + a/2^{n-m-1} \leqslant 1 \tag{4.4.22}$$

当 k_0 为偶数时，a 可取为偶数；k_0 为奇数时，a 可取为奇数。

证明：因为 $f(x)$是 m 阶相关免疫的，但不是 $m+1$ 阶相关免疫的，所以有

$$u: W_H(u) = m+1$$

使 $S_{(f)}(u)\neq 0$，不妨设 $u=(1,1,\cdots,1,0,\cdots,0)$，则

$$S_{(f)}(u)=\frac{1}{2^n}\sum_{x_1,x_2,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}\sum_{x_{m+1},\cdots,x_n}(-1)^{f(x)+x_{m+1}}$$

$$=\frac{1}{2^n}\sum_{x_1,x_2,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}\sum_{x_{m+1},\cdots,x_n}(1-2f(x))(-1)^{x_{m+1}}$$

$$=\frac{(-2)}{2^n}\sum_{x_1,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}\sum_{x_{m+1},\cdots,x_n}f(x)(-1)^{x_{m+1}}$$

$$=\frac{(-2)}{2^n}\cdot\sum_{x_1,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}\sum_{x_{m+1},\cdots,x_n}f(x)(1-2x_{m+1})$$

$$=\frac{(-2)}{2^n}\cdot\sum_{x_1,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}\sum_{x_{m+1},\cdots,x_n}f(x)+\frac{(-2)^2}{2^n}\cdot\sum_{x_1,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}$$

$$\sum_{x_{m+2},\cdots,x_n}f(x_1,\cdots,x_m,1,x_{m+2},\cdots,x_n)$$

因为 $f(x)$是 m 阶相关免疫的，由定理 4.4.11 可知，对任意固定的 $x_1,x_2,\cdots,x_m$，有

$$\sum_{x_{m+1},\cdots,x_n}f(x)=W_H(f)/2^m$$

因此

$$S_{(f)}(u)=\frac{(-2)^2}{2^n}\cdot\sum_{x_1,\cdots,x_m}(-1)^{\sum_{i=1}^{m}x_i}\sum_{x_{m+2},\cdots,x_n}f(x_1,\cdots,x_m,1,x_{m+2},\cdots,x_n)$$

依次类推，可得

$$S_{(f)}(u)=\frac{(-2)^{m+1}}{2^n}\left(\frac{W_H(f)}{2^m}-2\sum_{x_{m+2},\cdots,x_n}f(1,1,\cdots,1,x_{m+2},\cdots,x_n)\right)$$

令

$$a(u)=\frac{W_H(f)}{2^m}-2\sum_{x_{m+2},\cdots,x_n}f(1,1,\cdots,1,x_{m+2},\cdots,x_n)$$

$$=k_0-2\sum_{x_{m+2},\cdots,x_n}f(1,1,\cdots,1,x_{m+2},\cdots,x_n)$$

当 k_0 为奇数时，$a(u)$为奇数；当 k_0 为偶数时，$a(u)$为偶数，因此取 $a=|a(u)|$，a 为正整数，且

$$\max_{w\in F_2^n}|S_{(f)}(w)|\geqslant|S_{(f)}(u)|=a/2^{n-m-1}$$

由式(4.4.15)可得

$$N_f\leqslant 2^{n-1}(1-a/2^{n-m-1})$$

将此式整理后便可得式(4.4.22)。

式(4.4.22)表明，N_f 随 m 指数下降，相关免疫阶对非线性度的影响很大，在具体应用时，应适当折中。

设 $f(x):F_2^n\to F_2$ 的多项式表示为式(4.4.4)。现在我们用 $f(x)$的 Walsh 谱来表示式(4.4.4)中的系数。将由分量足标 $i_1i_2\cdots i_r$ 指定的 r 维及 $n-r$ 维子空间记为

$$S_{i_1i_2\cdots i_r}=\{x\in F_2^n|x_j=0,\text{对所有的 }j\text{ 在 }\{i_1,i_2,\cdots,i_r\}\}$$

$$\overline{S}_{i_1i_2\cdots i_r}=\{x\in F_2^n|x_j=0,\text{对所有的 }j\in\{i_1,i_2,\cdots,i_r\}\}=S_{i_1i_2\cdots i_r}^{\perp}$$

注意到式(4.4.4)中，除系数为 $a_{i_1i_2\cdots i_r}$的项之外，其余各项在 $S_{i_1i_2\cdots i_r}$上模 2 求和结果均为

零，因此有

$$a_{i_1i_2\cdots i_r}=\sum_{x\in\overline{S}_{i_1i_2\cdots i_r}}f(x)=\sum_{x\in\overline{S}_{i_1i_2\cdots i_r}}\left(\frac{1}{2}-\frac{1}{2}\sum_{w\in F_2^n}S_{(f)}(w)(-1)^{w\cdot x}\right)$$

$$=-\frac{1}{2}\sum_{w\in F_2^n}S_{(f)}(w)\sum_{x\in\overline{S}_{i_1i_2\cdots i_r}}(-1)^{w\cdot x}\qquad(\mathrm{mod}2)$$

$$=-\frac{1}{2}\sum_{w\in S^{\perp}_{i_1i_2\cdots i_r}}S_{(f)}(w)\cdot 2^r\qquad(\mathrm{mod}2)$$

$$=-2^{r-1}\sum_{w\in\overline{S}_{i_1i_2\cdots i_r}}S_{(f)}(w)\qquad(\mathrm{mod}2)\qquad(4.4.23)$$

$\overline{S}_{i_1i_2\cdots i_r}$中的 w 的重量 $W_H(w)\leqslant n-r$。

当 $f(x):F_2^n\to F_2$ 为 Bent 函数时，如果 $r\geqslant\frac{n}{2}+1$，即 $r-1\geqslant n/2$，由式(4.4.23)可知

$$a_{i_1i_2\cdots i_r}=-2^{r-1}\sum_{w\in\overline{S}_{i_1i_2\cdots i_r}}S_{(f)}(w)\quad(\mathrm{mod}2)$$

此时 $S_{(f)}(w)=\pm 2^{-\frac{n}{2}}$而且$|\overline{S}_{i_1i_2\cdots i_r}|=2^{n-r}$，所以 $a_{i_1i_2\cdots i_r}$必为偶数，故 $a_{i_1i_2\cdots i_r}=0(\mathrm{mod}2)$。这表明 Bent 函数的非线性次数不大于 $n/2$。

当 $f(x):F_2^n\to F_2$ 为 m 阶相关免疫函数时，如果 $r\geqslant n-m$，则根据定理 4.4.9，式(4.4.23)仅有 $S_{(f)}(0)$，于是 $a_{i_1i_2\cdots i_r}=-2^{r-1}S_{(f)}(0)(\mathrm{mod}2)$。又 $S_{(f)}(0)=2^{-n}(2^n-2W_H(f))$，所以当 $r\geqslant n-m$ 时

$$a_{i_1i_2\cdots i_r}=-2^{r-1}\times 2^{-n}(2^n-2W_H(f))(\mathrm{mod}2)=(2^{r-n+m}k_0-2^{r-1})(\mathrm{mod}2)$$

这里 $W_H(f)=2^mk_0$。所以当 $r>n-m$ 时，$a_{i_1i_2\cdots i_r}=0$。当 $r=n-m$ 时，若 k_0 为奇数，则所有的 $n-m$ 次项都出现；若 k_0 为偶数，则所有的 $n-m$ 次项都不出现。

当 $W_H(f)=2^{n-1}$，$1\leqslant m\leqslant n-2$ 时，易知，k_0 为偶数，于是对于 $r\geqslant n-m$ 都有 $a_{i_1i_2\cdots i_r}=0$。

综上所述，如果 $f(x):F_2^n\to F_2$ 是非线性次数为 k 的 m 阶相关免疫函数，则 $k+m\leqslant n$，特别地，当 f 是平衡布尔函数，$1\leqslant m\leqslant n-2$ 时，$k+m\leqslant n-1$。这表明 f 的非线性次数 $k=\partial f$ 和其相关免疫阶 m 之间也存在着某种制约关系，这对于非线性组合密钥流域生成器的设计有重要指导意义。目前，消除这种制约关系的办法有两种：一种是引进记忆[1,158]；另一种是引入广义相关免疫函数[78]。关于广义相关免疫函数的讨论可进一步参阅文献[35,74]。

本节我们讨论了布尔函数的一些非线性准则，我们已看到这些准则的一些是相互制约的，这就要求我们在具体构造时必须适当折中考虑。

值得注意的是，本节我们只讨论了一个输出的布尔函数，即 $f(x):F_2^n\to F_2$，但在某些应用场合需讨论多输出的布尔函数，即 $f(x):F_2^n\to F_2^m$，诸如下章将要介绍的 DES 的 S-盒就是一个多输出的布尔函数的实例。关于多输出的布尔函数的密码学性质的研究可参阅文献[35,75,76,77,78,79]。

4.5 构作流密码的四种方法

根据对密码分析者的能力和机会的假设的不同、密码分析成功的定义的不同以及安

全性的概念的不同，Rueppel[80]将构造流密码的方法分为四种：信息论方法，系统论方法，复杂度理论方法，随机化方法。下面我们来分别介绍这四种方法。

4.5.1 信息论方法

在密码的信息论方法的设计中，假定密码分析者具有无限的时间和计算能力。密码分析是指在只有给定密文以及各种密钥和消息的前验概率的条件下，确定消息或特定密钥的过程。我们说一个保密系统被破译是指密码分析者对每个给定的密文能找到在下列意义下"唯一的"一个解(消息)：该消息是解的概率本质上是1，别的消息是解的概率实际上是0。对敌手具有无限能力的假设暗含着在Shannon模型中，安全性的概念与加密和解密算法的复杂性是独立的。一个密码系统说是理想安全的(ideally secure)，如果对每一个给定的密文，不管密码分析者观察到多少密文，他也不能找到"唯一的"一个明文解。关于别的一些概念和讨论可参见本书第2章。

最著名的流密码系统之一是"一次一密"密码，它是目前唯一被证明具有完善保密性的系统。该系统的详细描述参见第2章。

我们知道，密钥流生成器的目的是将一个短的随机密钥(也称种子密钥)k扩展成一个长的伪随机串$Z^l=Z_1Z_2\cdots Z_l$，这可以简写为

$$G: K \to K^l \qquad Z^l = G(k)$$

以强调密钥k与密钥流Z^l之间的函数关系。

对一个长为n的二元密钥k^n和一个长为l的二元密钥流，(n,l)－序列生成器是使得$Z^l=G(k^n)$的一个从$\{0,1\}^n$到$\{0,1\}^l$的函数。

在复杂度理论中，一个生成器G被近似地定义为一个无限长的$(n,l_{(n)})$－序列生成器序列$\{G_n \mid n\geqslant 1\}$，这里$l$是$n$的一个多项式，每个序列生成器的计算时间的上界是$n$的一个多项式函数。

类似于密码系统的唯一解距离，我们可以定义密钥流生成器的唯一解距离，将其定义为在已知明文攻击下，确定密钥所需要的最少的密钥流符号的数目。在一个已知明文攻击中，因为关于明文没有不确定性，所以$H_L=0$。故由第2章中的有关讨论可知，二元密钥流生成器的唯一解距离$n_{G,u}\approx H(K)$。在Shannon模型下，当敌手获得不少于$n_{G,u}$长的密钥流比特时，无安全性可言。现在我们对Shannon模型作如下修改：假定敌手的观察是有限的，其观察量为e比特(未必连续)，且e小于$H(K)$。在这种修改的模型下，我们就可以谈论安全性问题，这就是所谓的局部随机化问题[81]。基于这种动机，Schnorr[82]提出了如下生成器：

Schnorr 伪随机生成器 $G=\{G_n \mid n\geqslant 1\}$

输入：种子密钥或实际密钥k(k是一个随机函数$f: I_m \to I_m$)和k的长度$m2^m$。

(1)置$y_i^0=i, i=0,1,\cdots,2^{2m}-1$；

(2)对$j=0,1,2$，计算

$$y_i^{j+1} = (R(y_i^j), L(y_i^j) \oplus f(R(y_i^j)))$$

($L(x)$和$R(x)$分别表示变量x的左半部分和右半部分的函数)。

输出：$G_n(k)$序列$y_i^3, i=0,1,\cdots,2^{2m}-1$($G_n(k)$的长度为$2m2^{2m}$)。

Schnorr生成器将一个长为$n=m2^m$的种子扩展成大约n^2长的伪随机比特串。

文献[83]使用系统论分析法，找到了从一段 $m2^m+O(m)$ 长的密钥流比特就能恢复密钥 k 的一种攻击方法，这种方法所需的时间是线性的。这说明 Schnorr 生成器不安全。这一结果表明，如果安全模型的假定和实际的应用不一致，那么安全性就降低了。使用 Shannon 模型用信息论术语直接定义安全性似乎是更自然的、更合理的。关于序列的随机化问题的进一步研究，感兴趣的读者可参阅文献[81]。

4.5.2 系统论方法

系统论方法是目前设计密码系统最广泛使用的一种实用设计方法。设计流密码的系统论方法有两个目标。一个目标是根据某些系统论度量指标诸如周期、线性复杂度、频率分布和到线性函数的距离等研制设计方法和具有可证明特性的模块。另一个目标是研究密码分析原理并建立一些使得基于这些原理的攻击变得不可行的设计准则。例如，这样的基本的密码分析原理有代替和线性逼近，分别征服攻击、统计分析等。为了阻止基于这些基本原理的密码分析，人们对密钥流生成器已提出了一系列的设计准则[1,16,55,67,68,84,85]，例如：

(1)长的周期。

(2)线性复杂度准则：大的线性复杂度，线性复杂度轮廓，局部线性复杂度，等等。

(3)统计准则：比如理想的游程分布。

(4)混淆：每个密钥流比特必须是所有或大多数密钥比特的一个复杂变换。

(5)扩散：子结构的冗余度必须扩散到大范围的统计中去。

(6)函数的非线性准则：相关免疫性，非线性度，雪崩准则，等等。

任何安全的密钥流生成器必须满足上述设计准则。在系统论方法中，可直接地设计出满足这些准则的流密码。

系统论方法不只是用来设计流密码，它还可用来设计分组密码，诸如 DES 就是一个典型的例子。

值得注意的是设计准则只能部分地反映一般的密码分析原理。一个满足所有设计准则的生成器也可能是不安全的。换句话说，这些设计准则是密钥流生成器安全的必要条件，而非充分条件。系统论方法的优点是每一个新的密码系统对密码分析者产生一个难的而且先前不知道的问题。当然同基于某些著名的问题诸如因子分解或离散对数的系统的密码分析相比，破译这种系统是不吸引人的。

下面我们介绍一大批利用系统论方法设计的密钥流生成器。

4.5.2.1 非线性滤波生成器和非线性组合生成器

从实用的观点来看，下列两类密钥流生成器是很重要的：

(1)非线性地滤波一个 LFSR 的状态(见图 4.5.1)；

(2)非线性地组合一些 LFSR 的输出(见图 4.5.2)。

前一种类型的生成器称作滤波生成器，函数 f_0 称作滤波函数，生成的序列称作滤波序列。后一种类型的生成器称作组合生成器，函数 f_0 称作组合函数，生成的序列称作组合序列。显然，滤波生成器是组合生成器的一种特殊情况。但是两种生成器的密码分析有点不同，所以我们仍然把它们分为两类生成器。这两类生成器中所选用的函数 f_0 必须是非

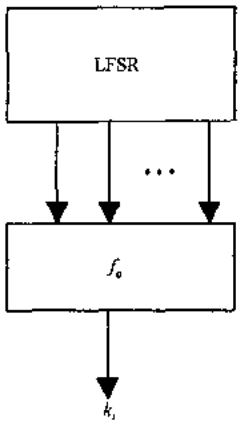


图 4.5.1 滤波生成器

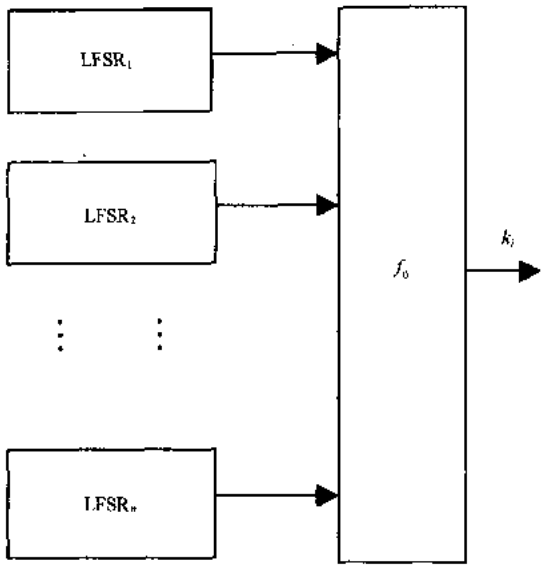


图 4.5.2 组合生成器

线性的，否则，滤波序列的线性复杂度不超过 LFSR 的级数，组合序列的线性复杂度不超过各个 LFSR 的级数之和。这样，所生成的序列的线性复杂度仍很低，不能抵抗基于 B-M 算法的攻击(即线性攻击)。

滤波生成器(filter generator)[86]

输入：参数：一个 n 级最大长 LFSR，反馈函数为 F_2 上的一个 n 次本原多项式 $f(x)$。
密钥：非线性输出函数 $f_0: F_2^n \to F_2$，LFSR 的初态 $\boldsymbol{a}_0$。

对 $i=1,2\cdots$，完成下列步骤：

(1)移位 LFSR(记状态变换为 f_1)：$\boldsymbol{a}_i=f_1(\boldsymbol{a}_{i-1})$；

(2)计算 $k_i=f_0(\boldsymbol{a}_i)$。

输出：序列$\{k_i|i\geqslant 1\}$。

文献[1,16,86,87,88]对滤波序列的线性复杂度等特性进行了分析。分析表明，该序列的线性复杂度与输出函数 f_0 的次数有关，现将其中一些结果不加证明地列出。但值得注意的是序列的迹表示和根表示是分析密钥流序列的线性复杂度、周期等特性的重要工具，关于这些讨论可参阅文献[1,6,7,16,89]。

定理 4.5.1 设滤波函数 f_0 的次数 $\partial^0 f_0=m$，LFSR 是 n 级最大长 LFSR，则：

(1)滤波序列的周期是 2^n-1 的因子。

(2)(key 界) 滤波序列的线性复杂度的上界为 $L_m=\sum_{i=1}^{m} C_n^i$。

(3)对一个固定的 n 级最大长 LFSR(n 为奇数)，在所有的非线性次数为 m 的滤波函数中，使滤波序列的线性复杂度达到最大 L_m 的滤波函数的概率为

$$p_m \approx \exp(-L_m/n \cdot 2^n) > e^{-1/n}$$

因此，对充分大的 n，$p_m\approx 1$，即大多数滤波生成器产生的序列的线性复杂度达到(2)中的上界 L_m。

(4)如果最大长 LFSR 的级数 n 是 4 的倍数，并且滤波函数 f 是一个 Bent 函数，则滤

波序列的线性复杂度的下界为 $L(k)\geqslant C_{n/2}^{n/4}\times 2^{n/4}$；

(5)设 $m<n$，$\gcd(s,2^n-1)=1$，n 元布尔函数 f_1 的非线性次数 $\partial^0 f_1<m$，取滤波函数为

$$f_0(x_1,x_2,\cdots,x_n)=\sum_{i=0}^{n-1}c_i x_i x_{i+s}\cdots x_{i+(m-1)s}+f_1(x_1,x_2,\cdots,x_n)$$

$c_i\in F_2(0\leqslant i\leqslant n-1)$，$c_i$ 不全为 0，则滤波序列的线性复杂度的下界为 $L(k)\geqslant C_n^m-(n-1)$。

(6)设滤波函数 $f=x_t x_{t+s}$，$1\leqslant t<t+s\leqslant n$，则滤波序列 $k\neq 0$ 的线性复杂度 $L(k)=\frac{n(n+1)}{2}$，周期为 2^n-1。

关于滤波生成器的攻击参阅文献[16,90,93]。

背包生成器[1](knapsack generator)是滤波生成器的一个具体实例。我们知道，背包问题是一个 NP-完全问题。文献[1]中基于背包问题提出了一类滤波生成器—背包生成器。

背包生成器

输入：参数：一个最大长 LFSR$\langle f(x),n\rangle$，模 Q。

密钥：一个 n 维背包向量$(w_1,\cdots,w_n)$，每个分量 w_i 的尺寸为 N 比特，LFSR 的初始状态 $x_0=x_{10}\cdots x_{n0}$。

对 $i=1,2,\cdots$，完成下列步骤：

(1)移位 LFSR；

(2)计算背包和 $s_i=\sum_{j=1}^{n}x_{ji}w_i \mod Q$ (将序列$\{s_i|i\geqslant 1\}$称为背包和序列)；

(3)抽取 s_i 的某一比特作为 k_i。

输出：序列$\{k_i|i\geqslant 1\}$。

因为 2^n-1 为背包和序列$\{s_i|i\geqslant 1\}$的周期，所以 2^n-1 也为第 j 比特序列 $\tilde{s}_j$ 的一个周期。文献[1]指出，如果 $Q=2^n$，则背包和序列$\{s_i|i\geqslant 1\}$的第 j 比特序列 $\tilde{s}_j$ 的线性复杂度的上界为

$$L(\tilde{s}_j)\leqslant\sum_{i=1}^{2^j}C_n^i \qquad j\leqslant[\log_2 n]$$

$$L(\tilde{s}_j)\leqslant\sum_{i=1}^{n}C_n^i=2^n-1 \qquad j>[\log_2 n]$$

这表明模 2^n 的背包的 $\log_2 n$ 个低位比特要比高位比特的密码学性质弱。

组合生成器(combination generator)

输入：参数：n 个 LFSR$\langle f_i(x),L_i\rangle$，非线性输出函数 $f_0:F_2^n\to F_2$。

密钥：n 个 LFSR 的初始状态 $a_0^{(j)}$，$j=1,2,\cdots,n$。

对 $i=1,2,\cdots$，完成下列步骤：

(1)对 $j=1,2,\cdots,n$，移位 LFSR$_j$ 并抽出 $a_i^{(j)}$；

(2)$k_i=f_0(a_i^{(1)},\cdots,a_i^{(n)})$。

输出：序列$\{k_i|i\geqslant 1\}$。

文献[1,87,91,92]对组合序列的线性复杂度和周期等特性进行了分析。文献[1]指出，组合序列的线性复杂度 $L(k)$ 和周期 $p(k)$ 的上界分别为 $L(k)\leqslant f(L_1,\cdots,L_n)$ 和 $p(k)\leqslant lcm(T_1,T_2,\cdots,T_n)$，其中 $T_i(1\leqslant i\leqslant n)$ 为 LFSR$_i$ 生成的序列的周期，L_i 为 LFSR$_i$ 的级

数。设 $f(x_1,x_2,\cdots,x_n)$的代数正规型为式(4.4.5)，$f(L_1,L_2,\cdots,L_n)$表示将式(4.4.5)中的 F_2 上的乘法和加法视作实数域上的加法和乘法进行计算所得的结果。在这里我们将不加证明地介绍一些关于组合序列的分析结果。

定理 4.5.2 设组合生成器中的 n 个 LFSR 都是最大长 LFSR，级数分别 $L_1,L_2,\cdots,L_n$，$L_i>2(1\leqslant i\leqslant n)$，并且对所有的 $i\neq j$，$\gcd(L_i,L_j)=1$，f 为输出函数，则组合序列的线性复杂度为 $L(k)=f(L_1,L_2,\cdots,L_n)$。

关于组合生成器的攻击参阅文献[16,67,94]。下面我们来介绍组合生成器的一些具体实例。

Geffe 生成器[95]

在组合生成器中，取 $n=3$，选 3 个最大长 LFSR 使得它们的级数两两互素，选组合函数 $f_0(x_1,x_2,x_3)=x_3+x_1x_2+x_2x_3=x_1x_2+\overline{x}_2x_3$，则所得的生成器即为 Geffe 生成器。该生成器的周期为 $T=T_1T_2T_3$，线性复杂度为 $L=L_3+L_1L_2+L_2L_3$，其中 T_i 和 L_i 分别为 $\mathrm{LFSR}_i(1\leqslant i\leqslant 3)$的最小周期和级数。虽然 Geffe 生成器生成的序列具有长的周期、大的线性复杂度、好的统计特性等许多优点，但该生成器是密码学上弱的，这是因为

$$p(f(x)=x_1)=p(f(x)=x_3)=\frac{3}{4}$$

关于该生成器的具体分析可参阅文献[96,97,98]。

Pless 生成器[99]

这种生成器是根据 J-K 触发器(J-K flip-flop)的性能而设计的，它是由驱动 4 个 J-K 触发器的 8 个 LFSR 构成，每一个 J-K 触发器作为其中两个 LFSR 的非线性组合器。四个触发器输出序列按步长 4 采样，然后交错地产生密钥流(如图 4.5.3 和 4.5.4)。一个 J-K 触发器可由函数 $y_j=x_j^{(1)}+y_{j-1}\overline{(x_j^{(1)}+x_j^{(2)})}$来表示。这里 y_j 表示时刻 j 的内部状态。

输入：参数：8 个 $\mathrm{LFSR}(f_i(x),L_i)$，$1\leqslant i\leqslant 8$。

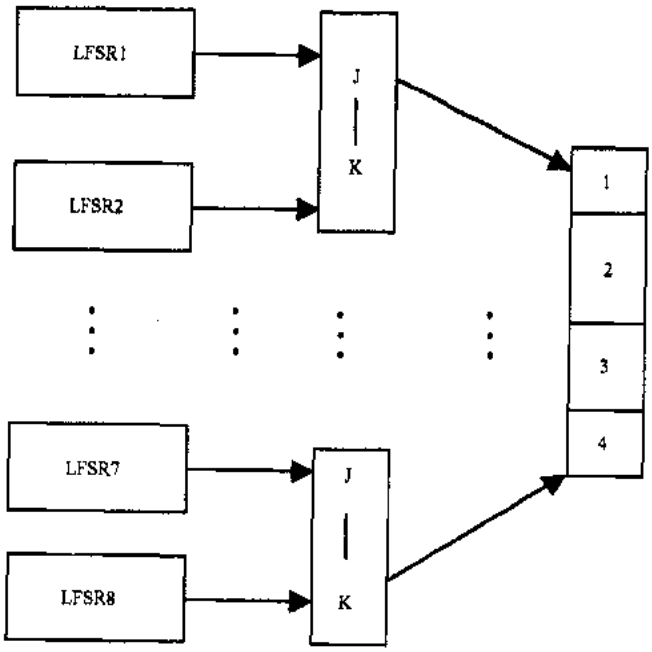


图 4.5.3 pless 生成器

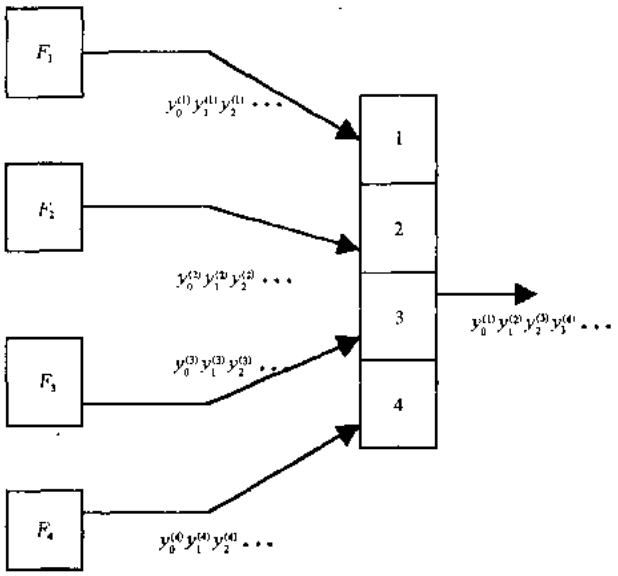


图 4.5.4 pless 生成器的简化表示

密钥：8 个 LFSR 的初始状态 $\boldsymbol{a}_0^{(1)},\cdots,\boldsymbol{a}_0^{(8)}$。

对 $i=0,1,2,\cdots$，完成下列步骤：

对 $k=1,2,3,4$，完成下列步骤：

(a)移位每一个 LFSR；

(b)对相应的一对 LFSR，计算第 k 个 J-K 触发函数

$$y_{4i}^{(k)}=a_{4i}^{(2k-1)}+y_{4i-1}^{(k)}(\overline{a_{4i}^{(2k-1)}+a_{4i}^{(2k)}})$$

(c)取四个密钥流比特 $k_{4i+k}=y_{4i}^{(k)}$。

输出：序列$\{k_l|l\geqslant 1\}$。

Pless 生成器是不安全的。关于该生成器的分析可参阅文献[100,101]。

门限生成器(threshold generator)[102]

这种生成器由 M 个 LFSR 和一个非线性组合规则 f 构成。

输入：参数：M 个最大长 LFSR$\langle f_i(x),L_i\rangle$，对任意 $i\neq j$，$\gcd(L_i,L_j)=1$。

密钥：M 个 LFSR 的初始状态 $\boldsymbol{a}_{10},\cdots,\boldsymbol{a}_{M0}$。

对 $i=1,2,\cdots$，完成下列步骤：

(1)对 $j=1,2,\cdots,M$，完成下列步骤：

(a)移位 LFSR_j；

(b)抽取 a_{ji}。

(2) 计算当前输入比特的整数和并确定

$$k_i=\begin{cases}1 & \sum_{j=1}^{m}a_{ji}>M/2\\ 0 & \text{否则}\end{cases}$$

输出：序列$\{k_i|i\geqslant 1\}$。

如果 M 为奇数，则该生成器的输出序列是平衡的。下面考虑 $M=3$，该时映射 f 能表示成

$$f(x_1,x_2,x_3)=x_1x_2+x_1x_3+x_2x_3$$

门限输出序列的周期 $T=T_1T_2T_3$，线性复杂度为 $L=L_1L_2+L_2L_3+L_1L_3$。虽然 $M=3$ 时的门限生成器的输出序列有较好的特性，但它仍是不安全的，因为对每一个 $j=1,2,3$，有 $H(f(x))-H(f(x)|x_j)=0.189$。

通常将门限生成器中的函数称为严格择多逻辑函数，关于这种函数的研究参阅文献[75,103,104]。

求和生成器(summation generator)[1]

在这种生成器中，将输入序列视作是某一整数的二进制表示，和函数 $f:Z^n\to Z$ 定义为 $f(x_1,x_2,\cdots,x_n)=\sum_{i=1}^{n}x_i$。

输入：参数：n 个 LFSR$(f_i(x),L_i)$，$1\leqslant i\leqslant n$。

密钥：n 个 LFSR 的初始状态和进位 C_0。

对 $i=1,2,\cdots$，完成下列步骤：

(1)移位每一个 LFSR_i，$1\leqslant i\leqslant n$；

(2)计算整数和 $S_i=\sum_{k=1}^{n}x_{ki}+C_{i-1}$；

(3)置 $k_i=S_i \bmod 2$，$C_i=\left[\frac{S_i}{2}\right]$。

输出：序列$\{k_i|i\geqslant 1\}$。

关于求和生成器的详细讨论见文献[1,105,106]。

4.5.2.2 钟控移位寄存器

钟控方法是另一种很诱人的设计密钥流生成器的方法。文献[107]是关于钟控移位寄存器的一篇很好的综述。本小节分两类钟控移位寄存器，即前馈钟控(forward clock control)和反馈钟控(feedback clock control)来介绍。基本的前馈钟控是指用一个规则的钟控移位寄存器来控制另一个移位寄存器的时钟。基本的反馈钟控是指用一个移位寄存器来控制它自己的时钟。

1. 前馈钟控生成器

基本的钟控生成器[80]

输入：参数：一个钟控 LFSR$(f_1(x),L_1)$，周期为 T_1，一个生成 LFSR$(f_2(x),L_2)$，周期为 T_2，映射 $f:F_2^{L_1}\to Z_{T_2}$。

密钥：两个 LFSR 的初始状态 $\boldsymbol{x}_0,\boldsymbol{y}_0,f$ 也可能是密钥。

对 $i=1,2,\cdots$，完成下列步骤：

(1)移位 LFSR_1 1 次并计算出 $f(x_i)$；

(2)移位 $\mathrm{LFSR}_2 f(x_i)$次并抽取诱导的输出 $y_{\sigma(i)}$，这里 $\sigma(i)=\sum_{k=1}^{i}f(x_k)$；

(3)置 $k_i=y_{\sigma(i)}$。

输出：序列$\{k_i|i\geqslant 1\}$。

当取 $f(x)=x_1$ 时，这种钟控生成器就是所谓的停走(stop-and-go)生成器[108,109]。停

走生成器是很不安全的[110]。当取 $f(x)=1+\sum_{j=0}^{n-1}x_j2^j$，$n\leqslant L_1$ 时，这种钟控生成器就是文献[111]中所讨论的钟控生成器，关于这种生成器的讨论和研究还可参阅文献[112，113，114]。

交错生成器(alternating step generator)[115]

这种生成器本质上是由共享同一钟控寄存器的两个停走生成器构成的，将对应的输出序列模 2 相加便得密钥流。

输入：参数：一个钟控 LFSR$\langle f_c(x),L_c\rangle$，周期为 T_c，一对生成 LFSR$\langle f_i(x),L_i\rangle$，$i=1,2$。

密钥：三个 LFSR 的初始状态 $x_0,y_0^{(1)},y_0^{(2)}$。

对 $i=1,2,\cdots$，完成下列步骤：

(1)移位 $LFSR_1$ 1 次并产生 x_i；

(2)如果 $x_i=1$，那么移位 $LFSR_1$ 并产生 $y_{\sigma(i)}^{(1)}$；如果 $x_i=0$，那么移位 $LFSR_2$ 并产生 $y_{i-\sigma(i)}^{(2)}$；

(3)置 $k_i=y_{\sigma(i)}^{(1)}+y_{i-\sigma(i)}^{(2)}$。

输出：序列$\{k_i|i\geqslant 1\}$。

交错生成器生成的序列具有很长的周期和很大的线性复杂度，关于这种钟控生成器的详细分析参见文献[115]。可用文献[97]中的方法攻破该生成器。

级联生成器(cascade generator)[117]

这种生成器由一串 LFSR 组成，每一个 LFSR 的时钟都受前一个 LFSR 的控制。

输入：参数：一个基本的 LFSR$\langle f(x),L\rangle$，周期为 T_0，级联的 LFSR 的个数为 N。

密钥：$S_0^{(n)}(n=1,2,\cdots,N)$的初始状态。

对 $i=1,2,\cdots$，完成下列步骤：

(1)移位第一个 LFSR 并产生 $y_i^{(1)}$；

(2)对 $n=2,\cdots,N$，完成下列步骤：移位第 n 个 LFSR$y_i^{(n-1)}$ 次(也可以移位第 n 个 LFSR $(y_i^{(n-1)}+1)$次)并产生

$$y_i^{(n)}=y_i^{(n-1)}+S_{\sigma_{n-1}(i)}^{(n)}$$

$$\sigma_{n-1}(i)=\sum_{k=1}^{i}y_i^{(k-1)}$$

(3)置 $k_i=y_i^{(N)}$。

输出：序列$\{k_i|i\geqslant 1\}$。

关于级联生成器的详细讨论请参阅文献[109，112，116，117，118]。

收缩生成器(shrinking generator)[119]

这种生成器由两个 LFSR 构成。通过用一个 $LFSR_1$ 选择另一个 $LFSR_2$ 的输出来生成密钥流(见图 4.5.5)。

输入：参数：两个 LFSR$\langle f_i(x),L_i\rangle$，$i=1,2$。

密钥：两个 LFSR 的初始状态 $a_0^{(1)},a_0^{(2)}$。

对 $i=1,2,\cdots$，完成下列步骤：

(1)移位 $LFSR_1$ 并产生 $y_i^{(1)}$；

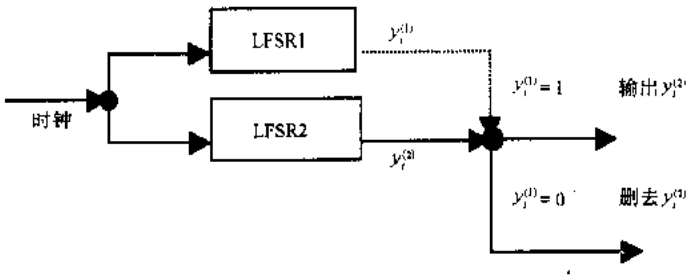


图 4.5.5 收缩生成器

(2)移位 $LFSR_2$ 并产生 $y_i^{(2)}$;

(3)如果 $y_i^{(1)}=1$,则置 $k_i=y_i^{(2)}$;如果 $y_i^{(1)}=0$,则删去 $y_i^{(2)}$。

输出:序列$\{k_i|i\geqslant 1\}$。

文献[119]对这种生成器生成的序列的线性复杂度、周期等性质进行了研究。文献[120]引入另一种钟控生成器,即自收缩生成器,文献[120]也指出了这两种生成器的等价性。文献[121,123,124]给出了这两类生成器的一些攻击方法,这些方法表明,收缩生成器和自收缩生成器似乎都是不安全的。

2. 反馈钟控生成器

$[d,k]$-自采样生成器[122]

输入:参数:一个 LFSR$(f(x),L)$,周期为 T_0,步规则 $f:\{0,1\}\to Z_{T_0}$。

密钥:LFSR 的初始状态 y_0。

对 $i=0,1,2,\cdots$,完成下列步骤:

(1)析出 $Z_i=y_{\sigma(i)}$;

(2)移位 LFSR$f(y_{\sigma(i)})$次并更新 $\sigma(i)$,$\sigma(i+1)=\sigma(i)+f(y_{\sigma(i)})$。

输出:序列$\{Z_i|i\geqslant 1\}$。

关于钟控生成器的分析,还可参阅文献[125,126,127,128]。

除了上述介绍的生成器外,还有一些生成器诸如多路复合生成器、内积生成器、细胞自动机生成器、$1/p$ 生成器等。这些生成器大都不安全,限于篇幅,这里就不再逐一介绍,感兴趣的读者请参阅文献[80,129]。

虽然上面所介绍的生成器大都有这样或那样的安全性问题,但这使我们可以从中吸取很多经验和教训,这对我们用系统论方法设计好的生成器具有重要的指导意义。

4.5.3 复杂度理论方法

在复杂度理论模型中,所有的计算都是一个安全参数,通常是密钥长度的函数并且所完成的分析也是近似的。只有运行时间是输入尺寸的多项式的算法才认为是计算上可行的。而且假定敌手具有多项式时间的攻击能力。密码分析是指(1)预测密钥流的一个数字,或者(2)区分一个密钥流序列和一个真正随机的序列的过程。如果一个密钥流生成器是(1)不可预测的,或者(2)与真正的随机生成器是多项式时间不可区分的,我们就说该密钥流生成器是完全的。但这种完全生成器是一个假想的器件,并不知道它是否存在。目前提出的生成器都是基于某些著名数学问题的困难性的假设,诸如离散对数、二次剩余和因子分解等问题。这些生成器的"完全性"并不是真正意义上的完全性,而是一个相对的概念。

不过我们仍延用完全性这一概念。下面我们来详细地讨论这些问题。

4.5.3.1 基本概念

定义 4.5.1 一个比特生成器 G 是一列多项式时间算法$\{G_n|n\geqslant 1\}$。每一个 $G_n:\{0,1\}^n\to\{0,1\}^{l}$ 将一个长度为 n 的随机密钥 x^n 扩展成一个长度为 $l_{(n)}$的伪随机序列 Z^l，这里 l 是 n 的一个多项式函数。设 $Z^l=G_n(x^n)$表示由 G_n 关于输入 x^n 产生的密钥流。

设 $\mu_{R,l}$表示 l 比特序列集上的均匀概率分布，即 $\mu_{R,l}(s^l)=2^{-l}$。对应地，设 $\mu_{G,l_{(n)}}$表示由 G_n 关于随机选择的密钥(即密钥根据 $\mu_{R,n}$来选择)产生的密钥流序列 Z^l 的概率分布，则

$$\mu_{G,l}(Z^l)=2^{-n}|\{x^n\in\{0,1\}^n|G_n(x^n)=Z^l\}|$$

设 μ_G 是由 G 产生的概率分布序列$\{\mu_{G,l_{(n)}}|n\geqslant 1\}$，$\mu_G$ 说成是由 G 诱导的概率分布序列。

对一个密钥流生成器的一个实际的安全要求是它的不可预测性。给定 i 比特长的一个子段将它扩展成大于 i 比特长的序列必须是不可行的。否则，截获密钥流的一个子段无需密钥的知识就能解密一部分密文。不可预测性的概念可通过下一比特测试(next-bit test)或预测器(predictor)的概念来正式地定义。

一个预测器(下一比特测试)$C=\{C_n|n\geqslant 1\}$是一列概率多项式规模的电路，每一个 C_n 具有 $i_n<l_{(n)}$个输入门和一个二元输出门。我们说一个预测器 C 可预测一个伪随机生成器 G。如果存在一个多项式 $p(n)$使得对无限多的 n，有

$$p(C_n(Z^i)=Z_{i+1})\geqslant\frac{1}{2}+\frac{1}{p(n)}$$

这里 Z^l 根据 $\mu_{G,l_{(n)}}$来抽取。这时，我们也说 G 没能通过下一比特测试 C。我们说 G 通过 C，如果对充分大的 n 和所有多项式 $p(n)$

$$p(C_n(Z^i)=Z_{i+1})<\frac{1}{2}+\frac{1}{p(n)}$$

我们说 G 是不可预测的，如果对所有的下一比特测试 C，对充分大的 n，对所有的多项式 $p(n)$，以及所有的 $i<l_{(n)}$有

$$p(C_n(Z^i)=Z_{i+1})<\frac{1}{2}+\frac{1}{p(n)}$$

一个密钥流生成器试图有效地模拟随机性。如果它产生的伪随机序列和真正的随机序列是有效地可区分的，那么人们就不能声称该生成器模拟随机性。不可区分的概念可通过统计测试的概念来正式地定义。

对下一个比特生成器的一个统计测试 $T=\{T_n|n\geqslant 1\}$是一列概率多项式规模的电路，每一个 T_n 具有 $l_{(n)}$个输入门和一个二元输出门。我们说一个测试 T 可区分 G，如果存在一个多项式 $p(n)$使得对无限多的 n，有

$$|p_n^{T,G}-p_n^{T,R}|\geqslant\frac{1}{p(n)}$$

这里 $p_n^{T,G}$ 和 $p_n^{T,R}$ 分别表示 T 根据 $\mu_{G,l_{(n)}}$和 $\mu_{R,l}$抽取的序列作为输入输出 1 的概率。我们说一个生成器通过统计测试 T，如果对所有的多项式 $p(n)$和充分大的 n，有

$$|p_n^{T,G}-p_n^{T,R}|<\frac{1}{p(n)}$$

文献[130]指出，预测器和统计测试这两个概念是等价的，即一个比特生成器 G 通过

所有的下一比特测试 C 当且仅当它通过所有的统计测试 T。

称一个比特生成器 G 是完全的，如果它通过所有的多项式规模的统计测试 T。不幸的是，已提出的生成器中没有一个能被证明是完全的。甚至我们，不知道完全的生成器是否存在。文献[131]在文献[130]的基础上，提出了构造比特生成器的一个一般方案。设 $f=\{f_n|x_n\to x_n\}$ 是一个用作生成器的下一状态函数的单向置换，所谓单向置换就是给定 x，计算 $f(x)$ 是容易的，但给定 $f(x)$，要找 x 却是难的。设 $B=\{B_n|x_n\to\{0,1\}\}$ 是定义在 x_n 上的用作输出函数的一个二元关系。随机地选择一个元素 $x\in x_n$ 作为种子，关于 x 迭代 f_n，输出

$$Z_i = B_n(f_n^i(x)) \qquad 1\leqslant i\leqslant l_{(n)}$$

如果 B 对 f 是一个不可预测的关系，那么密钥流 Z^l 中的比特对左边的比特是不可预测的。由文献[131]中的结论可知，密钥流 Z^l 对任何统计测试是不可区分的(特别地它对右边的比特也是不可预测的)。下面我们来讨论这个方案的一些具体例子。

4.5.3.2 伪随机比特生成器

Shamir 伪随机数生成器[132]

输入：参数：模 $N=pq$，N 的长度为 n，一个固定的序列 $e_1,e_2,\cdots,e_l$(称为公钥指数)，对每一个 $i(1\leqslant i\leqslant l)$，$\gcd(e_i,\varphi(N))=1$(其中 $\varphi(N)$表示欧拉函数)，随机地从 Z_N 中选取一个种子 S 并公开。

密钥：因子 p 和 q。

对 $i=1,2,\cdots,l$，计算

$$d_i = 1/e_i \bmod\varphi(N)\text{(称 } d_i \text{ 为解密指数)}$$
$$Z_i = S^{d_i} \bmod N$$

输出：序列$\{Z_i|i\geqslant 1\}$。

这个生成器输出 n 比特数，不只是 1 个比特。只有知道因子 p 和 q，才能计算出 $d_i=1/e_i\bmod\varphi(N)$。不知道因子 p 和 q，计算 $d_i=1/e_i \bmod \varphi(N)$在计算上是难的。文献[132]证明了预测这种伪随机数生成器的输出等价于破译 RSA 体制。文献[131,133]也对这种生成器作了一些讨论。RSA 体制将在本书第 6 章中介绍。

Blum-Micali 生成器[131]

设 p 是一个奇素数，α 是 Z_p^* 的一个生成元。一个元素 $y\in Z_p^*$ 关于 α 的离散对数定义为使得 $y=\beta^x \bmod p$ 的唯一的整数 x，$0\leqslant x\leqslant p-2$，表示为 $x=\log_\alpha y$。离散对数问题就是关于输入 p,α,y 寻找 $x=\log_\alpha y$。一般地，没有有效的算法被知道能解决离散对数问题。如果 $y\in QR_p$，即 y 是模 p 的一个二次剩余，那么对某一整数 $t<(p-1)/2$，有 $x=\log_\alpha y=2t$。$y\in QR_p$ 的两个平方根是 $\alpha^t \bmod p$ 和 $\alpha^{t+(p-1)/2} \bmod p$，$\alpha^t \bmod p$ 称为 y 的主平方根(principal square root)，$\alpha^{t+(p-1)/2} \bmod p$ 称为 y 的非主平方根(nonprincipal square root)。对每一个 p 和 α，$x\in Z_p^*$ 是一个主平方根，当且仅当 $\log_\alpha x<(p-1)/2$。文献[131]证明了确定一个给定的元素是否是关于 p 和 α 的主平方根的有效算法的存在暗含着计算离散对数的有效算法的存在。

定义二元关系

$$\text{half}_p(x)=\begin{cases}1 & x<\dfrac{p-1}{2}\\ 0 & \text{否则}\end{cases}$$

Blum-Micali 生成器

输入：参数：(p,α)。

密钥：一个随机选择的种子 $x_1\in Z_p^*$。

对 $i=1,2,\cdots,l$，完成下列步骤：

(1)$Z_i=\text{half}_p(x_i)$；

(2)$x_{i+1}\leftarrow\alpha^{x_i} \bmod p$

输出：序列$\{Z_i|1\leqslant i\leqslant l\}$。

该生成器的安全性基于计算模 p 的离散对数的困难性。

RSA 生成器[134]

取 $f(x)=x^e \bmod N$ 作为一个单向置换，$B(x)=lsb(x)$作为对 f 不可预测的关系。设 N 是一个长度为 n 的整数且 $N=pq$，$e\geqslant 3$ 是一个整数且 $\gcd(e,\varphi(N))=1$。

输入：参数：(N,e)。

密钥：一个随机选择的种子 $x\in Z_N^*$。

(1)$x_0\leftarrow x$；

(2)对 $i=1,2,\cdots,l$，完成下列步骤：

(a) 计算 $x_i\leftarrow x_{i-1}^e \bmod N$；

(b) 抽取 $Z_i\leftarrow lsb(x_i)$($lsb(x_i)$表示 x_i 的最低位比特)。

输出：序列$\{Z_i|1\leqslant i\leqslant l\}$。

该生成器的安全性基于破译 RSA 体制的困难性。

二次剩余生成器[135](也称 Blum-Blum-Shub 生成器，简称 BBS 生成器)

设 N 是两个不同的奇素数 p 和 q 之积。一个元素 $y\in Z_N^*$ 称作一个模 N 的二次剩余，如果对某一 $x\in Z_N^*$，有 $y=x^2 \bmod N$。用 QR_N 表示模 N 的二次剩余之集。每一个 $y\in QR_N$ 恰有 4 个平方根。如果 $p\equiv q\equiv 3 \bmod 4$，那么这 4 个平方根中恰有一个平方根属于 QR_N。因此，映射 $x\in QR_N\rightarrow x^2 \bmod N\in QR_N$ 是一个双射。其逆映射为

$$y\in QR_N\rightarrow \sqrt{y} \bmod N\in QR_N$$

文献[136]证明了在下述意义下分解 N 和计算平方根是等价的问题：对这两个问题之中的一个问题存在有效算法暗含着对另一个问题也存在有效算法。

二次剩余生成器

输入：参数：模 N，N 的长度为 n。

密钥：一个随机选择的 $x_1\in QR_N$(当然 p 和 q 也保密)。

对 $i=1,2,\cdots$，完成下列步骤：

(1)$Z_i=lsb(x_i)$；

(2)$x_{i+1}=x_i^2 \bmod N$。

输出：序列$\{Z_i|1\leqslant i\leqslant l\}$。

如果 $p\equiv q\equiv 3 \bmod 4$，那么在 Z_N^* 中恰好有一半元素的 Jacobi 符号为 $+1$，另一半元素的 Jacobi 符号为 -1，分别记为 J_N^{+1}和 J_N^{-1}。J_N^{-1}中的元素都不是二次剩余，而 J_N^{+1}中恰好有

一半元素是二次剩余。二次剩余问题就是关于输入 N 和 $x\in J_N^1$ 确定 x 是否是模 N 的一个二次剩余。寻找一个有效的算法解决二次剩余问题还是一个公开问题。二次剩余生成器的安全性是基于确定二次剩余的困难性。文献[135]证明了关于输入 N 和 $y\in J_N^1$ 确定二次剩余可以有效地归结为确定 $x=\sqrt{y}\bmod N$ 的最低比特。

关于用复杂度理论研究随机比特生成器的讨论可参阅文献[137]的第十二章，该章对一些基本概念和一些有代表性的生成器作了比较详细而清楚的解释和论证，尤其对初学者值得一读。

4.5.4 随机化方法

一般地，要证明解决一个给定问题的所有或几乎所有实例所做的计算努力的下界是困难的。但是我们可以用一种变换的方法来考虑问题，不去考虑解决一个给定问题所做的工作努力，而是考虑有待解决的密码分析问题的规模。设计的目标是增加密码分析者在密码分析过程中所要检查的比特量而尽可能保持秘密密钥量小。这可通过在加密和解密过程中使用长的公开可获得的随机串来实现。密钥将指定长的随机串中的哪一部分用来加密和解密，而不知道密钥的敌手不得不对随机串作穷搜索。这种类型的密码的安全水平是通过在密码分析者也许能够采用比纯粹猜测更好的办法确定密钥或明文之前，他不得不检查的平均比特数来衡量的。为了使密码分析者有一个不可忽略的成功概率，设计的目标将是对密码分析者必须完成的比特测试的期望值建立一个可证明的下界。这个下界可能有不同的解释。如果比特测试的期望值的下界是任何算法破译该系统所必须完成的步骤数的下界，这就是计算安全性的概念。如果比特测试的期望值的下界也是在敌手的各种密钥和消息的后验概率改进之前，他不得不观察的比特数的下界，这就是信息论意义下安全的概念。下面我们来给出一些基于随机化方法的流密码，不过这些流密码都不实用。

Diffie 的随机化流密码[80]

输入：消息 $x=x_1x_2\cdots$。

密钥 k：一个随机的 n 比特串。

(1) 产生 2^n 个随机序列 $r_1,r_2,\cdots,r_{2^n}$；

(2) 使用第 k 个随机串 r_k 作为一次本加密 x。

输出：在公开的信道上将 $y=x\oplus r_k$ 和 $r_1,r_2,\cdots,r_{2^n}$ 发送给接收者。

Rip Van Winkle 密码[138]

输入：明文序列 $\tilde{x}=x_1x_2\cdots$。

密钥：一个随机的 n-比特数 k，$0\leqslant k<2^n$。

(1)产生一个随机的 k-比特串 $r_1,r_2,\cdots,r_k$；

(2)产生一个随机的密钥流 $\tilde{Z}=Z_1Z_2\cdots$；

(3)对 $i=1,2,\cdots$，完成下列步骤

(a) 用随机密钥流 $\tilde{Z}$ 将明文序列 $\tilde{x}$ 加密成密文序列 $\tilde{y}^{(1)}$：

$$\tilde{y}_i^{(1)}=x_i\oplus Z_i$$

(b) 按下列方法形成第二个序列 $\tilde{y}^{(2)}$：

$$\tilde{y}_i^{(2)} = \begin{cases} r_i & 1 \leqslant i \leqslant k \\ Z_{i-k} & i > k \end{cases}$$

(4) 交替地发送 $\tilde{y}^{(1)}$和 $\tilde{y}^{(2)}$的比特。

输出：比特对序列$\{(y_i^{(1)}, y_i^{(2)}) | i \geqslant 1\}$。

Maurer 的随机化流密码[139]

输入：明文 $x^N=(x_1,\cdots,x_N)\in F_2^N$，公开一个随机函数发生器 $R[s,t]$，$1\leqslant s\leqslant S$，$0\leqslant t\leqslant T-1$。

密钥：$k^S=(k_1,\cdots,k_S)\in Z_T^S$。

(1)计算密钥流

$$Z_i = \sum_{s=1}^{S} R[s,(k_s+i-1)\bmod T]\bmod 2 \qquad 1\leqslant i\leqslant N$$

(2)加密 $y^N=x^N\oplus Z^N$。

输出：密文 y^N。

4.6 注记和文献

流密码理论相对而言比较成熟，主要原因之一是数学这一工具可用于流密码的研究。有限域和移位寄存器序列是流密码的基础，关于移位寄存器，可参阅文献[5,6,7,8,11]。关于有限域，可参阅文献[5,6]。目前已有几本流密码专著，感兴趣的读者可参阅文献[1,16,67,75]。当然，一些密码学著作诸如文献[129,137,141,142]也用专门的章节介绍了流密码。

我们在各节中介绍有关内容时，已指出了它们的出处及一些有关消息，对有关问题感兴趣的读者可根据这些线索进一步查阅资料。

我们知道，流密码生成器有多种多样，每种生成器都可能有好几种攻击方法，目前已有大量的攻击流密码的方法，但最有代表性的方法有以下几种：分别征服(divide and conquer)攻击[67]、最佳仿射逼近(best affine approximation)攻击[16,75]，线性校验子(linear syndrome)攻击[97]和线性一致性测试(linear consistency test)攻击[140]。限于篇幅，我们不能在本书中详细介绍这些攻击方法，但这些攻击方法很值得一读。

对多输出函数、有记忆的前馈网络密码系统、非线性逼近感兴趣的读者可参阅文献[1,77,78,79,143,144,145,158]，这些话题国外学者非常关注。

关于 M 序列的构作，除了文献[75]所提到的文献外，文献[146]也给出了一种很好的构作方法。

关于环上的序列的研究，感兴趣的读者可参阅文献[147～155]。

值得一提的是虽然本章中我们介绍了大量的流密码生成器，但大部分都不是很安全的，希望使用者在选用时慎重，必须弄清所选用的生成器是否安全，以及近来的发展情况。

参 考 文 献

[1] Rueppel, R., Analsis and Design of Stream Ciphers, Springer-Verlag, 1986.

[2] Campbell, C. M., Design and Sepcificaition of Cryptographic Capabilites, IEEE Comm. Soc. Mag. Vol. 16(6)pp. 15—19(Nov. 1978).

[3] Gait, J., A New Nonliear Pseudorandom Number Generator, IEEE Trans, on software Eng. Vol. SE—3(5) pp. 359—363(Sept. 1977).

[4] Diffie, W. And Hellman, M., Privacy an Authentication, An Introuction to Cryptography, Proc. IEEE Vol. 67(3) pp. 397—427(Mar. 1979).

[5] Lidl, R. and Niederreiter, Introduction to Finite Fields and Their Applications, Cambridge University Press, 1986.

[6] 万哲先,代数和编码,科学出版社,北京,1985.

[7] 肖国镇,梁传甲,王育民,伪随机序列及其应用,北京,国防工业出版社,1985.

[8] Golomb, S. W., Shift Register Sequences, San Francisco: Holden-Day, 1967.

[9] Massey, J. L., Shift-Register Synthesis and BCH Decoding, IEEE Trans. IT, Jan. 1969,122—127.

[10] Berlekamp, E. R. Algbric Coding Theory, New York: McGraw-Hill, 1968.

[11] 万哲先,戴宗铎,刘木兰,冯绪宁,非线性移位寄存器,科学出版社,北京,1978.

[12] Solomonov, R. T., A formal Theory of Inductive Inference, Part I, Inform. Control 7, 1964.

[13] Kolmogorov, A. N., Three Approaches to the Quantitative Defmition of Information, Probl, Inform. Transmission, Vol. q, No. 1, pp. 3—11(in Russian), 1965.

[14] Lempel, A. And Ziv, J., On the Complexity of Finite Sequences, IEEE Trans. On IT-22, No. 1, pp. 75—81, Jan. 1976.

[15] Games, R. A. and Chan, A. H., Afast Algorithm for Determining the Complexity of a Binary Sequence with Period 2^n, IEEE Trans., 1983, IT-29:pp. 144—146.

[16] Ding, C., Xiao, G. and Shan, W., The Stability Theory of Stream Ciphers, Springer-Verlay, New York, 1991.

[17] 陈克非,序列的 d-复杂度,电子学报,1989,17(1),pp. 112—113.

[18] Niederreiter, H., The Linear Complexity Profile and the Jump Complexity of Keystream Sequences, Advances in Cryptology-Eurocrypt'90, Springer-Verlay, 1991,pp. 174—188.

[19] Jansen, C. J. A. and Boekee, D. E., The Shortest Feedbanck Shift Register That Can Generate a Given Sequence, Advances in Cryptology-Crypto'89, Springer-Verlay, 1990,pp. 90—99.

[20] Chan, A. H. and Games, R. A., On the Quadratic Spans of Periodic Sequences, Advances in cryptology-crypto'89, Springer-Verlay, 1990, pp. 82—89.

[21] 沈世镒,组合密码学,杭州,浙江科学技术出版社,1992.

[22] Yang Junhui and Dai Zongduo, Linear Complexity of Periodically Repeated Random Sequences, Acta Mathematica Sinica, New Series 1995, Vol. 11, Special Issue (July), pp. 1—7.

[23] Dornstetter, J. L. On the Equivalence between Berlekamo's and Euclid's Algorithms, IEEE Trans., May 1987, IT-33 (3).

[24] Cheng, U., On the Continued Fraction and Belekamp's Algorithm, IEEE Trans. Znform. Theory, May 1984, IT-30,pp. 541—544.

[25] Dai, Z. D. and Zeng, K., Continued Fractions and Berlekamp-Massey Algorithm, Advances in Cryptology-Auscrpt'90, Springer-Verlag, 1990, pp. 24—31.

[26] Schaub, T., A Linear Complexity Approach to Cyclic codes, Ph. D Thesis, Swiss Federal Institute of Technology, ADAG Administration & Druch AG, Zurich, 1988.

[27] Blahut, R. E., Transform Techniques for Error-Controll Codes, IBM, J. Res. Devel., 1979, 23,pp. 299—315.

[28] Massey, J. L. and Serconek, S., A Fourier Transform Approach to the Linear Complexity of Nonlinearly Filtered Sequences, Advances in Cryptology-Crypto'94, Springer-Verlag, 1949, pp. 332—340.

[29] Massey, J. L. and Serconek, S., Linear Complexity of Periodic Sequences: A General Theory, Advances in Cryptology-Crypto'96, Springer-Verlag, 1996,pp. 359—372.

[30] 武传,多值逻辑函数与其多元的几种无关性的谱分析,电子科学学刊,1993,Vol. 15(No. 1), 17—25 页.

[31] 张木想、肖国镇,多值逻辑函数相关免疫的谱特征,科学通报,1994,Vol. 39(No. 9), 772—773 页.

[32] 李世取、曾本胜，多值逻辑函数相关免疫的充要条件，密码学进展-chinacrypt'94，科学出版社，北京，1994，257—264 页.

[33] 冯登国，环 Z_N 上的相关免疫函数的频谱特征，电子学报，1997，Vol. 25 (No. 7)，115—116 页.

[34] 冯登国、肖国镇，有限域上的函数的相关免疫性和线性结构的谱特征，通信学报，1997，Vol. 18(No. 1)，40—45 页.

[35] 冯登国，频谱理论及其在通信保密技术中的应用[博士论文]，西安电子科技大学，西安，1995.

[36] 冯登国、裴定一、肖国镇，密码学中的多值逻辑的研究，自然科学进展，Vol. 8，No. 3，1998，257—261 页.

[37] Gopalakrishnon, K. and Stinson, D. R., Three Characterizations of Non-binary Correlation-immune and Resilient Functions, Designs, Code and Cryptography, 1995, 5: pp. 241—251.

[38] 冯登国、裴定一，关于多值逻辑函数的仿射逼近，电子科学学刊，Vol. 19，No. 5，1997，713—716 页.

[39] Camion, P. and Canteaut, A., Generalization of Siegenthaler Inequality and Schnorr-voudenay Multipermutations, Advances in Cryptology-Crypto'96, Berlin, Springer-Verlag, 1996, pp. 372—386.

[40] 刘福运，Reed-Muller 变换及其在流密码中的应用[硕士论文]，西安电子科技大学，西安，1988.

[41] Karpovsky, M., Finite Othogonal Series in the Design of Digital Devices, New York, John Wiley & Sons, 1976.

[42] Rothaus, O. S., On Bent Functions, J. Combinatorial Theory (Ser. A), 1976, 20: pp. 300—305.

[43] Carlet, C., Two New Classes of Bent Functions, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, 1994, pp. 77—101.

[44] 张木想、肖国镇，关于 Bent 函数与其变元的非线性组合之间的相关性，科学通报，Vol. 19(No. 19)，1994.

[45] 冯登国、肖国镇，Bent 函数与其变元的相关特性，电子学报，Vol. 24，No. 11，1996.

[46] Macwilliams, F. J. and Sloane, N. J. A., The Theory of Error-Correcting Codes, North Holland Publishing Company, 1977.

[47] 武传坤，密码学中的布尔函数[博士论文]，西安电子科技大学，西安，1993.

[48] Seberry, J., Zhang, X. M. and Zheng, Y., On Constructions and Nonlinearity of Corretation Immune Functions, Adrances in Cryptology-Eurocrypt'93, Springer-Verlag, 1994, pp. 181—199.

[49] Seberry, J., Zhang, X. M. and Zheng, Y., Nonlinearly Balanced Bolean Functions and Their Propagation Characteristic, Advances in Cryptology-Crypto'93, Springer-Verlag, 1994, pp. 49—60,

[50] Seberry, J., Zhang, X. M, Highly Nonlinear 0—1 Balanced Boolean Functions Satisfying Strict Avalanche Criterion, Advances in Cryptology-Auscrypt'92, Springer-Verlag, 1993, pp. 145—155.

[51] Patterson, N. J. and Wiedeman, D. H., The Covering-radius of the (2^{15},16) Reed-Muller Code Is at Least 16 276, IEEE Transactions on Information Theory, IT-29 (3), pp. 354—356, 1983.

[52] Cohen, G. D. et al., Covering Radius-survey and Recent Results, IEEE Trans. Inform. Theory, Vol. IT-31, No. 3, 1985, pp. 328—343.

[53] 周锡君、陈卫红，布尔函数的 Walsh 变换的推广及布尔函数的非线性逼近，密码学进展—Chinacrypt'92，科学出版社，216—221 页.

[54] 冯登国、李春祥、肖国镇，关于布尔函数的二次逼近，通信学报，Vol. 15，No. 4，1994，34—38 页.

[55] Meier, W. and Staffebach, O., Nonlinearity Criteria for Cryptographic Functions, Advances in Cryptology-Eurocrypt'89, Springer-Verlag, 1990, 549—562.

[56] 李斌，具有线性结构 Boolean 函数的结构与计数，密码与信息，No. 1，1995，1—5 页.

[57] 冯登国、肖国镇，布尔函数的对偶性和线性点，通信学报，Vol. 17，No. 1，1996，46—50 页.

[58] Webster, A. F. and Tavares, S. E., On the Design of S-boxes, Advances in Cryptology-Crypto'85, Springer-Verlag, 1986, pp. 523—534.

[59] Forrè, R., The Strict Avalanche Criterion: Spectras Properties of Boolean Functions and an Extended Definiton, Advances in Cryptology-Crypt'88, Springer-Verlag, 1990, pp. 450—468.

[60] Lloyd, S. A., Characterising and Counting Functions Satisfying the Strict Avalanche Criterion of Order (n-3), Proceedings of the Second IMA Confference on Cryptography and Coding, 1989.

[61] 廖勇，广义严格雪崩准则及满足它的布尔函数的性质，密码与信息，No. 3，1993，5—12 页.

[62] Cusick, W., Boolean Functions Satisfying a Higher Order Strict Avalanche Criterion, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, 1994, pp. 86－95.

[63] Prenceel, B., Govaerts, R. and Vandewalle, J., Boolean Functions Satisfying Higher Order Propagaton Criteria, Advances in Cryptology-Eurocrypt'91, Springer-Verlag, 1992, pp. 141－152.

[64] Carlet, C., Partially Bent Functions, Advances in Cryptology-Crypto'92, Springer-Verlag, 1993, pp. 280－291.

[65] Preneel, B., Vanleekwijk, W., Van Linden, L., Govaerts, R. and Vandewalle, J., Propagation Characteristics of Boolean Functions, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 161－173.

[66] 冯登国、肖国镇，满足k次扩散准则的布尔函数的谱特征，电子学学刊，Vol. 18, No. 4, 1996, 385－390 页.

[67] Siegenthaler, T., Methoden fur den Entwurf Von Stream Cipher System, ADAG Administration, Durch AG. lurich, 1986.

[68] Xiao, G. Z. and Massey, J. L., A Spectral Charaterization of Correlation-Immunity Combining Functions, IEEE Trans. Informat. Theory, Vol. IT34(3), pp. 569－571, May 1988.

[69] 冯登国、肖国镇，对偶距离和相关免疫阶，通信学报，Vol. 15, No. 1, 15－16 页, 1994.

[70] Siegenthaler, T., Correlation-Immunity of Nonlinear Combing Functions for Cryptolgraphic Applications, IEEE Trans. Informat. Theory, Vol. IT-30, No. 5 Sept. 1984, pp. 776－780.

[71] 单炜娟，相关免疫函数的结构与构造，应用数学报，Vol. 14, No. 3, 1991, 331－336 页.

[72] 何良生，相关免疫布尔函数之特征，密码与信息，No. 4, 1990, 18－26 页.

[73] 杨义先、胡正名，抗熵漏前馈网络研究，电子学学刊，Vol. 13, No. 3, 1991, 232－240 页.

[74] 张木想、肖国镇，无记忆的组合函数的非线性与相关免疫性，电子学报，Vol. 22, No. 7, 1994.

[75] 丁存生、肖国镇，流密码学及其应用，国防工业出版社，北京，1994.

[76] Pieprzyk, J. And Finkelstein, G., Towards Effective Nonlinear Cryptosystem Design, IEE Proceedings, Pt. E, 135 (6), pp. 325－335, 1988.

[77] Nyberg, K., On the Construction of Highly Nonlinear Permutations, Advances in Cryptology-Eurocrypt'92, Springer-Verlag, 1993, pp. 92－98.

[78] Zhang, X. M. and Zheng, Y., On Nonliear Resilient Functions, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, 1995, pp. 274－288.

[79] Kurosawa, K. And Satoh, T., Generalization of Higher Order SAC to Vector Output Boolean Funtions, Advances in Cryptology-Eurocrypt'96, Springer-Verlag, 1996, pp. 218－231.

[80] Rueppel, R. A., Stream Ciphers, in Contemporary Cryptology: The science of Information Integrity, G. J. Simmons, ed., IEEE Press, 1992, pp. 65－134.

[81] Maurer, U. And Massey, J. L., Perfect Local Randomness in Pseudo-random Sequences, Advances in Cryptology-Eurocrypt'89, Springer-Verlag, 1990, pp. 100－112.

[82] Schnorr, C. P., On the Construction of Random Number Generators and Random Function Generators, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1988, pp. 225－232.

[83] Rueppel, R. A., On the Security of Schonorr's Pseudo Rnadom Sequence Generator, Advances in Cryptology-Eurocrypt'89, Springer-Verlag, 1990, pp. 423－428.

[84] Shannon, C. E., Communication Theory of Secrecy Systems, Bell Syst. Tech. J., Vol. 28, pp. 656－715, Oct. 1949.

[85] Beker, H. And Piper, F., Cipher Systems: the Protection of Communications, London, Northwood, Books, 1982.

[86] Groth, E. J., Gencration of Binary Sequences with Controllable Complexity, IEEE Trans. Inform. Theory, Vol. IT-17, No. 3, May 1971, pp. 288－296.

[87] Key, E. L., An Analysis of the Structure and Complexity of Nonlinear Binary Sequence Generators, IEEE Trans. Inform. Theory, Vol. IT-22, No. 6, pp. 732－763, Nov. 1976.

[88] Kumar, P. V. and Scholtz, R. A., Bounds on Linear Span of Bent Sequeces, IEEE Trans. Inform. Theory, Vol. IT-29, pp. 854－862, Nov. 1983.

[89] Herlestam, T., On the Complexity of Functions of Linear Shift Registers, Advances in Cryptology-Eurocrypt'85, Springer-Verlag, 1986, pp. 119—129.

[90] Siegenthaler, T., Cryptanalysts Representation of Nonlinearity Filtered ML-Sequences, Advances in Cryptology-Eurocrypt'85, Springer-Verlag, 1986, pp. 103—110.

[91] Golic, J. D., On the Linear Complexity of Funcitons of Periodic GF(q)-sequeces, IEEE Trans. Inform. Theory, Vol. IT-35, pp. 69—75, Jan. 1989.

[92] Brynielsson, L. On the Linear Complexity of Combined Shift Register Sequences, Advances in Cryptology-Eurocrypt'85, Springer-Verlag, 1986, pp. 156—166.

[93] Forre, R., A fast Correlation Attack on Nonlinearly Feedforward Filtered Shift Register Sequences, Advances in Cryptology-Eurocrypt'89, Springer-Verlag, 1990, pp. 586—595.

[94] Meier, W. and Staffelbach, O., Fast Correlation Attacks on Stream Ciphers, J. Cryptology, Vol. 1, No. 3, pp. 159—176, 1989.

[95] Geffe, P. R., How to Protect Data with Ciphers That Are Really Hard to Break, Electronics, Jan. 4, 1973, pp. 99—101.

[96] Zeng, K. And Huang. M, On the Linear Syndrome Method in Cryptanalysis, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1990, pp. 469—478.

[97] Zeng, K., Yang, C. H. and Rao, T. R. N., An Improved Linear Syndrome Algorithm in Cryptanalysis with Applications, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 34—47.

[98] 曾肯成，密码体制中的熵漏现象，中国科技大学研究生院资料，1986.

[99] Pless, V. S., Encryption Schemes for Computer Confidentiality, IEEE Trans. Comput. Vol. C-26, pp. 1133—1136, Nov. 1977.

[100] Rubin, F., Decrypting a Stream Cipher Based on J-K flip-flops, IEEE Trans. Comput., Vol. C-28, No. 7, pp. 483—487, July 1979.

[101] 黄民强，Pless 体制的相关性缺陷及攻击，中国科学技术大学研究院资料，1986.

[102] Bruer, J. O., On Pseudo Random Sequences as Crypto Generators, in Proc. Int. Zurich Seminar on Digital Communication, Switzerland, 1984.

[103] Titsworth, R. C., Optimal Ranging Codes, IEEE Trans. On Space Electronics and Telemetry, March 1964, pp. 19—30.

[104] 冯登国、肖国镇，严格择多逻辑函数的密码学特征，电子科学学刊，Vol. 15, No. 6, Nov. 1993, 643～646 页.

[105] Dawson, E. and Clark, A., Divide and Conquer Attacks on Certain Classes of Stream Ciphers, Cryptologia, January, 1994, No. 1, pp. 25—40.

[106] Stffebach, O. and Meier, W., Cryptographic Singificance of the Carry for Ciphers Based on Integer Addition, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 601—614.

[107] Gollmann, D. and Chambers, W. G., Clock-Controlled Shift Registers: A Review, IEEE Journal on Selected Areas in Communications, Vol. 7, No. 4, May 1989, pp. 525—533.

[108] Beth, T. and Piper, F., The Stop-and-Go Generator, Advances in Cryptology-Eurocrypt'84, Springer-Verlag, 1985, pp. 99—92.

[109] Vogel, R., On the Linear Complexity of Cascaded Sequences, Advances in Cryptology-Eurocrypt'84, Springer-Verlag, 1985, pp. 99—109.

[110] Zeng, K. C., Yang, C. H., Wei, D. Y. and Rao, T. R. N., Pseudorandom Bit Generators in Stream-Cipher Cryptography, IEEE Computer, Vol. 24, No. 2, Feb 1991, pp. 8—17.

[111] Chambers, W. G. and Jennings, S. M., Linear Equi Valence of Certain BRM Shift-register Sequences, Electron. Lett., Vol. 20, Nov. 1984.

[112] Tretter, S. A., Properties of PN^2 Sequences, IEEE Trans. Inform. Theory, Vol. IT-20, pp. 295—295, March 1974.

[113] Smeets, B., A Note on Sequences Generated by Clock-controlled Shift Registers, Advances in Cryptology-

Eurocrypt'85, Springer-Verlag, 1986, pp. 40—42.

[114] Golić, J. and Zivković, M. V., On the Linear Complexity of Nonuniformly Decimated pn-Sequences, IEEE Trans. Inform. Theory, Vol. 34, pp. 1077—1079. Sept. 1988.

[115] Günther, C. G., Alternating Step Generatiors Controlled by de Bruijn Sequences, Advances in Cryptology-Eurocrypt'87, Springer-Verlag, 1988, pp. 5--14.

[116] Chambers, W. G. and Gollmann, D., Generators for Sequences with Nearmaximal Linear Equi Valence, IEE, Proc. E., Vol. 135, pp. 67--69, 1988.

[117] Gollman, D. Pseudo Random Properties of Cascade Connections of Clock Controlled Shift Registers, Advances in Cryptology-Eurocrypt'84, Springer-Verlag, 1985,pp. 93—98.

[118] Gollman, D., and Chamger, W. G., Lock-ineffct in Cascades of Dock-conrtolled Shift-registers, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1988, pp. 331—343.

[119] Coppersmith, D., Krawczys, H. and Mansour, Y., The Sharinking Generator, Advances in Cryptology-Crypto'93, Springer-Verlag, 1994, pp. 22—39.

[120] Meier, W. and Staffelbach, O., The Self-Shrinking Generator, Advances in Cryptology-Eurocrypt'94, Springer-Verlag, 1995,pp. 205--214.

[121] 张遵法、陈伟东，关于对 Shrinking Generator 及 Self-Shrinking Generator 的熵漏分析，通信学报，Vol. 17, No. 4, July, 1996,pp. 15—20.

[122] Ruepple, R. A., When Shift Registers Clock Themselves, Advances in Cryptology-Eurocrypt'87, Springer-Verlag, 1988, pp. 13—15.

[123] Golić, J., Intrinsic Statistical Weakness of Key-stream Generators, Advances in Cryptology-Eurocrypt'94, Springer-Verlag, 1995, pp. 91—103.

[124] Mihaljević, M. J., A Security Examination of the Self-shrinking Generator, Presentation at 5th IMA Conference on Cryptography and Coding, Cirencester, U. K., December 1995.

[125] Gollmann, D., Cryptanalysis of Clock Controlled Shift Registers, Fast Software Encryption, Springer-Verlag, 1994,pp. 121—126.

[126] Golić, J., Towards Fast Correlation Attacks on Irregularly Clocked Shift Registers, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, 1995, pp. 248—262.

[127] Golić, J. and Mihaljević, M., A Generalized Correlation Attack on a Class of Stream Ciphers Based on the Levenshtein Distance, Journal of Cryptology, 3(1991), pp. 201—212.

[128] Golić, J. and U'Connor, L., Embedding and Probabilistic Correlation Attacks on Clock-controlled Shift Registers, Advances in Cryptology-Eurocrypt'94, Springer-Verlag, 1995, pp. 230—243.

[129] Schneier, B., Applied Cryptography : Protocols, Algorithms, and Source Code in C, John Wiley & Sons, New York, 2nd Edition, 1996.

[130] Yao, A. C., Theory and Applications of Trapdoor Functions, Proc. 25th IEEE Symp. Foundations Comput. Sci., New York, 1982.

[131] Blum, M., and Micali, S., How to Generate Cryptographically Strony Sequences of Pseudo-random Bits, SIAM J. Comput., Vol. 13, pp. 850—864,1984.

[132] Shamir, A., On the Generation of Cryptographically Strong Pseudo-random Sequences, 8th Int. Colloquium on Automatu, Languages, and Programmming, Springer-Verlag, 1981.

[133] Schnorr, C. P. and Alexi, W., RSA-bits Are 0.5+secure, Advances in Cryptology-Eurocrypt'84, Springer-Verlag, 1985, pp. 113—126.

[134] Alexi, W., Chor, B., Goldreich, O. and Schnorr, C. P., RSA and Rabin functions: Certain Parts Are as Hard as the Whole, SIAM J. Comput, Vol. 17, pp. 194—209, April 1988.

[135] Blum, L., Blum, M. and Shub, M., A Simple Unpredictable Pseudo-random Number Generator, SIAM J. Comput., Vol. 15, pp. 364—383, 1986.

[136] Rabin, M. O., Digital Signatures and Public-key Functions as Intractable as Factorization, Masachusetts Institute

of Techology Laboratory for Computer Science, TR-212, 1979.

[137] Stinson, D. R., Cryptography: Theory and Practice, CRC Press, Boca Raton, Florida, 1995.

[138] Massey, J. L. and Ingemarsson, I., The Rip Van Winkle Cipher-a Simple and Provably Computationally Secure Cipher with a Finite key, IEEE Int. Symp. Inform. Theory, Brighton, England, June 24—28 1985.

[139] Maurer, U., A Provavle-secure Strongly-randomized Cipher, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 361—373.

[140] Zeng, K., Yang, C. H. and Rao, T. R. N., On the Linear Consistency Test (LCT) in Cryptanalysis with Applications, Advances in Cryptology-Eurocrypt'89, Springer-Verlag, 1990, pp. 164—174.

[141] 王育民、何大可，保密学-基础与应用，西安电子科技大学出版社，西安，1990.

[142] 杨义先、林须端、胡正名，编码密码学，人民邮电出版社，北京，1992.

[143] Xiao, G. Z. and Zhang, M. X., Maximum Correlation Analysis of Nonliear Combinitg Functions, Proceedings of the Information Theory, Japan, 1995.

[144] 冯登国、裴定一、肖国镇，非线性组合函数的最大相关分析，中国科学(E 辑)，Vol. 28，No. 3，1998，238—243 页.

[145] 龚奇敏、黄月江，无记忆组合函数的相关度，电子学报，Vol. 19，No. 4，July 1991，40—46 页.

[146] Yang, J. H. and Dai, Z. D., Construction of m-ary de Bruijn sequences, Advances in Cryptology-Eurocrypt'92, Springer-Verlag, 1993, pp. 357—363.

[147] Dai, Z. D. and Huang, M. Q., A Criterion for Primitiveness of Polynomials over $Z/(2^d)$, Chinese Science Bulletin, Vol. 36(11), pp. 892—895, 1991.

[148] Huang, M. Q., Maxinmal Period Polynomials over $Z/(p^d)$, Science in China, (series A) Vol. 35(3), pp. 270—275, 1992.

[149] Dai, Z. D. and Huang, M. Q., Linear Complexity and The Minimal Polynomial of Linear Recurring Sequences over $Z/(m)$, Syst. Sci. and Math. Sci., Vol. 4(1), pp. 51—54, 1991.

[150] Pinch, R. G. E., Recurrent Sequences Modulo Prime Powers, The Inst. Of Math. & Its Appl. Conf. Ser., Crypt. And CodingIII, New Ser. No. 45, pp. 297—310, 1993.

[151] Dai, Z., Beth, T. and Gollmann, D., Lower Bounds for the Linear Complexity of Sequences over Residue Rings, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 189—195.

[152] Dai, Z. D., Binary Sequences Derived from ML-Sequences over Rings, I: Periods and Minimal Ploynomials, Journal of Cryptoloty, 5(1992), pp. 193—207.

[153] Qi, W. F. and Zhou, J. J., Polynomial Splitting Ring and Root Representation of Linear Recurring Sequences over, Science China(series A) $Z/(p^e)$, Vol. 37(9), pp. 1047—1052, 1994.

[154] 李献刚，流密码体制的研究与分析 [博士论文]，西安电子科技大学，1995.

[155] 周锦君、戚文峰、周玉洁，Gröbner 基推广及 $Z/(m)$ 上多条序列综合算法，中国科 学(A 辑)，Vol. 25，No. 2，1995，113—120 页.

[156] Nyberg, K., Construction of Bent Functions and Difference Sets, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1990, pp. 151—160.

[157] Yarlagadda, R. and Hershey, J. E., Analysis and Synthesis of Bent Sequences, Proc. IEE. Vol. 136, pt. E. pp. 112—123, March 1989.

[158] Meier, W. And Staffelbach, O., Correlation Properties of Combiners with Memory in Stream Ciphers, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1990, pp. 204—213.

第 5 章 私钥密码算法——分组密码

本章主要介绍私钥分组密码算法的基本概念、设计原则和工作模式以及现有的一些有代表性的私钥分组密码算法例如 DES、IDEA、RC5 等。

5.1 分组密码的设计原则

分组密码是将明文消息编码表示后的数字序列 $x_1, x_2, \cdots$ 划分成长为 m 的组 $x=(x_1, x_2, \cdots, x_m)$，各组(长为 m 的向量)分别在密钥 $k=(k_1, k_2, \cdots, k_l)$ 的控制下变换成等长的输出数字序列 $y=(y_1, y_2, \cdots, y_n)$(长为 n 的向量)，分组密码的模型如图 5.1.1 所示。它与流密码的不同之处在于输出的每一位数字不是只与相应时刻输入明文数字有关，而是与一组长为 m 的明文数字有关。分组密码有其自身的优点，首先分组密码容易被标准化，因为在今天的数据网络通信中，信息通常是被成块地处理和传输的。其次，使用分组密码容易实现同步，因为一个密文组的传输错误不会影响其它组，丢失一个明密文组不会对其随后的组的解密的正确性产生影响。分组密码的主要缺陷表现在两个方面：一是分组加密不能隐蔽数据模式，即相同的密文组蕴含着相同的明文组；二是分组加密不能抵抗组的重放、嵌入和删除等攻击。但分组密码的上述缺陷可以通过在加密处理中引入少量的记忆来克服。例如可以通过 5.4 节中所介绍的密码分组链接(CBC)模型来克服这些缺陷。

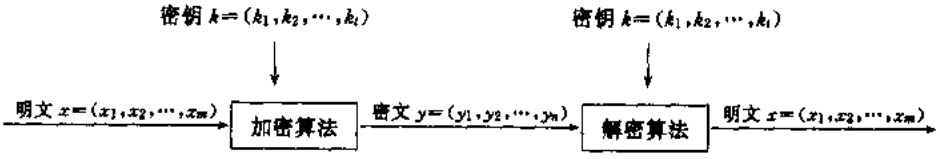


图 5.1.1 分组密码模型

若 $n>m$，则它为有数据扩展的分组密码。若 $n<m$，则为有数据压缩的分组密码。若 $n=m$，则为无数据扩展和压缩的分组密码，通常研究的均为这种情况。在本章中，我们主要讨论二元情况，也就是明文 x 和密文 y 均为二元数字序列，它们的每个分量 $x_i, y_i \in \{0, 1\}$。设 F_2 是一个二元域，F_2^s 表示 F_2 上的 s 维向量空间。假定明文空间和密文空间均为 F_2^m，密钥空间为 S_K，S_K 是 F_2^t 的一个子集合。m 是明文组和密文组以比特形式出现的长度，称为分组长度，t 是秘密密钥以比特形式出现的长度，称为密钥长度。

一个私钥分组密码可定义如下：

定义 5.1 一个私钥分组密码是一种满足下列条件的映射 $E: F_2^m \times S_K \to F_2^m$：对每个 $k \in S_K$，$E(\cdot, k)$ 是从 F_2^m 到 F_2^m 的一个置换。将一个分组密码简记为 $Y=E(X, K)$。

通常称 $E(\cdot, k)$ 为密钥 k 下的加密函数，称 $E(\cdot, k)$ 的逆为密钥 k 下的解密函数，记为 $D(\cdot, k)$。分组密码的真正的密钥规模被定义为 $l=\log_2|S_K|$ 比特。因而，密钥长度等于

真正的密钥规模当且仅当 $S_K=F_2^l$。例如下节将要介绍的 DES 算法的真正的密钥长度仅为 $l=56$ 比特。

我们知道，F_2^m 上的置换共有 $2^m!$ 个。由定义 5.1 可知，一个分组密码是 F_2^m 上的全体置换所构成的集合的一个子集合。可见，设计分组密码的问题在于找到一种算法，能在密钥控制下从一个足够大且足够"好"的置换子集合中简单而迅速地选出一个置换，用来对当前输入的明文数字组进行加密变换。一个好的分组密码应该是既难破译又容易实现的，加密函数 $E(\cdot,k)$和解密函数 $D(\cdot,k)$都必须是很容易计算的，但是要从方程 $y=E(x,k)$和 $x=D(y,k)$中求出密钥 k 应该是一个困难问题。下面我们将从安全性原则和实现原则两个方面来介绍分组密码的设计原则。

分组密码的设计原则

(1)针对安全性的一般设计原则。影响安全性的因素很多，诸如分组长度 m 和密钥长度 t 等。但有关实用密码的两个一般的设计原则是 Shannon 提出的混乱原则和扩散原则[1]。

混乱：人们所设计的密码应使得密钥和明文以及密文之间的依赖关系相当复杂以至于这种依赖性对密码分析者来说是无法利用的。

扩散：人们所设计的密码应使得密钥的每一位数字影响密文的许多位数字以防止对密钥进行逐段破译，而且明文的每一位数字也应影响密文的许多位数字以便隐蔽明文数字统计特性。

(2)针对实现的设计原则。分组密码可以用软件和硬件来实现。硬件实现的优点是可获得高速率，而软件实现的优点是灵活性强、代价低。基于软件和硬件的不同性质，分组密码的设计原则可根据预定的实现方法来考虑。

软件实现的设计原则：使用子块和简单的运算。密码运算在子块上进行，要求子块的长度能自然地适应软件编程，比如 8、16 和 32 比特等。在软件实现中，按比特置换是难于实现的，因此我们应尽量避免使用它。子块上所进行的一些密码运算应该是一些易于软件实现的运算，最好是用一些标准处理器所具有的一些基本指令，比如加法、乘法和移位等。

硬件实现的设计原则：加密和解密可用同样的器件来实现。尽量使用规则结构，因为密码应有一个标准的组件结构以便其能适应于用超大规模集成电路实现。

乘积密码是实现 Shannon 提出的混乱原则和扩散原则的一种有效的方法。所谓乘积密码就是两种或两种以上简单密码的逐次应用。由合理选择的许多子密码构成的乘积密码既可实现良好的混乱又可实现良好的扩散。下节将要介绍的 DES 就是一种乘积密码。

5.2 数据加密标准(DES)

计算机通信网的发展对信息的安全保密的要求日益增长，未来的数据传输和存贮都要求有密码保护。为了实现同一水平的安全性和兼容性，提出了数据加密标准化。标准化便于联网、训练操作维护人员、降低生产成本和推广使用。为此，美国商业部所属国家标准局(NBS-National Bureau of Standards)在 1972 年开始了一项计算机数据保护标准的发展

规划。NBS 在 1973 年 5 月 13 日的联邦记录(FR1973)中公布了一项公告，征求在传输和存贮数据中保护计算机数据的密码算法的建议，这一举措最终导致了数据加密标准(DES)的研制。DES[2]是迄今为止世界上最为广泛使用和流行的一种分组密码算法，它是由美国 IBM 公司研制的，是早期的称作 Lucifer 密码[3,4]的一种发展和修改。DES 在 1975 年 3 月 17 日首次被公布在联邦记录中，在做了大量的公开讨论后于 1977 年 1 月 15 日正式批准并作为美国联邦信息处理标准，即 FIPS-46，同年 7 月 15 日开始生效。规定每隔五年由美国国家保密局(NSA-National Security Agency)作出评估，并重新批准它是否继续作为联邦加密标准。最近的一次评估是在 1994 年 1 月，美国已决定 1998 年 12 月以后将不再使用 DES。美国目前正在征集、评估和制定新的数据加密标准，新的标准被称作 AES。尽管如此，DES 对于推动密码理论的发展和应用起了重大作用，对于掌握分组密码的基本理论、设计思想和实际应用仍然有着重要的参考价值。下面我们来描述这一算法。

5.2.1 DES 的描述

DES 的一个详细而清楚的介绍参见文献[5]，它是一个分组密码算法。DES 算法使用长度为 56 比特的密钥加密长度为 64 比特的明文，获得长度为 64 比特的密文，其加密工作程序如下。

(1)给定一个明文 x，通过一个固定的初始置换 IP 置换 x 获得 x_0，记 $x_0=IP(x)=L_0R_0$，这里 L_0 是 x_0 的前 32 比特，R_0 是 x_0 的后 32 比特。

(2)然后进行 16 轮完全相同的运算，在这里数据与密钥相结合。我们根据下列规则计算 L_iR_i，$1\leqslant i\leqslant 16$：

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$$

这里⊕表示两个比特串的异或，f 是一个函数(f 将在下面描述)，$k_1, k_2, \cdots, k_{16}$都是密钥 k 的函数，长度均为 48 比特(实际上，每一个 k_i 是来自密钥 k 的比特的一个置换选择)，k_1，$k_2, \cdots, k_{16}$构成了密钥方案。一轮 DES 加密过程如图 5.2.1 所示。

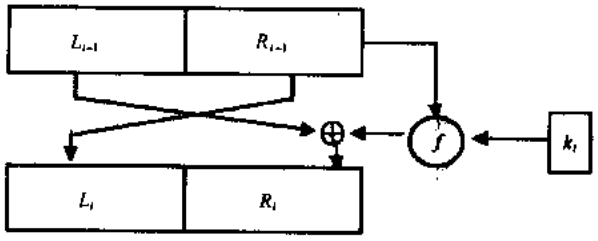


图 5.2.1 一轮 DES 加密过程

(3)对比特串 $R_{16}L_{16}$应用初始置换 IP 的逆置换 IP^{-1}，获得密文 y，即 $y=IP^{-1}(R_{16}L_{16})$。注意最后一次迭代后，左边和右边未交换，而将 $R_{16}L_{16}$作为 IP^{-1}的输入，目的是为了使算法可同时用于加密和解密。

函数 $f(A,J)$的第一个变量 A 是一个长度为 32 的比特串，第二个变量 J 是一个长度为 48 的比特串，输出是一个长度为 32 的比特串。f 的计算过程如下：

(1)将 f 的第一个变量 A 根据一个固定的扩展函数 E 扩展成一个长度为 48 的比特串。

(2)计算 $E(A)\oplus J$，并将所得结果分成 8 个长度为 6 的比特串，记为 $B=B_1B_2B_3B_4B_5B_6B_7B_8$。

(3)使用 8 个 S-盒 $S_1,S_2,\cdots,S_8$。每一个 S_i 是一个固定的 4×16 阶矩阵，它的元素来自 0～15 这 16 个整数。给定一个长度为 6 的比特串，比方说 $B_j=b_1b_2b_3b_4b_5b_6$，我们按下列办法计算 $S_j(B_j)$：用两个比特 b_1b_6 对应的整数 $r(0\leqslant r\leqslant3)$来确定 S_j 的行(所谓两个比特 b_1b_6 对应的整数 r 意指 r 的二进制表示为 b_1b_6，以下的含义类同)，用四个比特 $b_2b_3b_4b_5$ 对应的整数 $c(0\leqslant c\leqslant15)$来确定 S_j 的列，$S_j(B_j)$的取值就是 S_j 的第 r 行第 c 列的整数所对应的二进制表示。记 $C_j=S_j(B_j),1\leqslant j\leqslant8$。

(4)将长度为 32 的比特串 $C=C_1C_2C_3C_4C_5C_6C_7C_8$ 通过一个固定的置换 P 置换，将所得结果 $P(C)$记为 $f(A,J)$。函数 f 描述在图 5.2.2 中。

下面我们来描述 DES 算法中所使用的具体函数和密钥方案的计算。

初始置换 IP 和其逆置换 IP^{-1}如下：

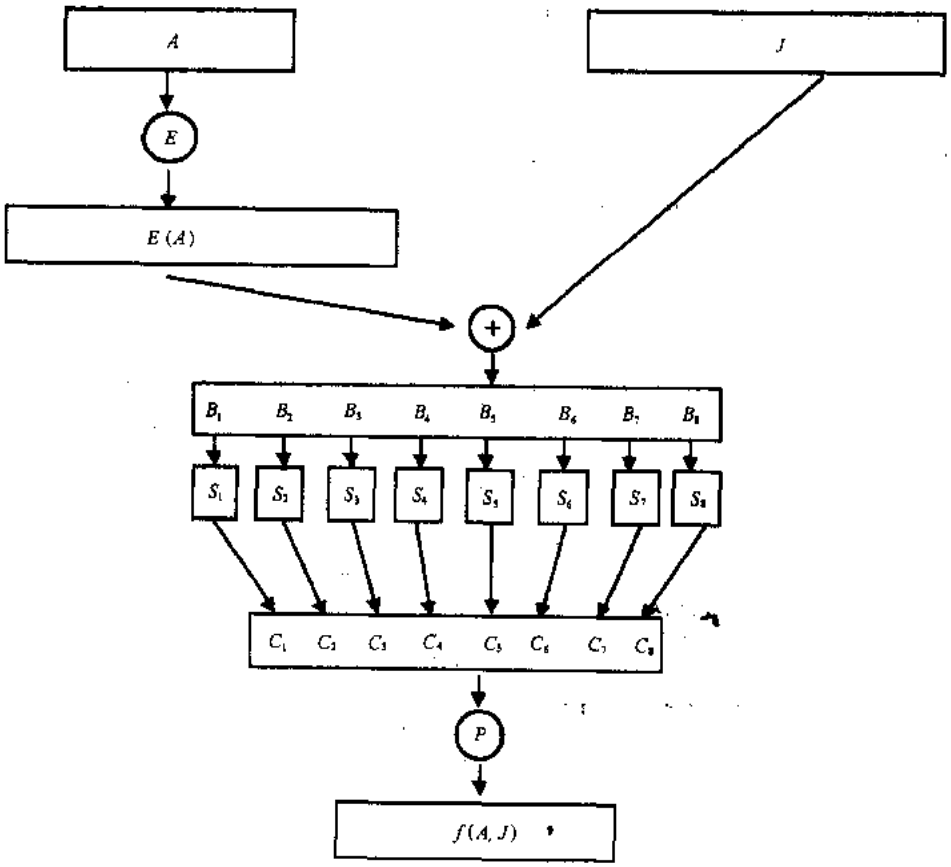


图 5.2.2 DES 的 f 函数

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

这意味着 x 的第 58 比特是 $IP(x)$的第 1 比特，x 的第 50 比特是 $IP(x)$的第 2 比特等等。初始置换 IP 及其逆置换 IP^{-1}没有密码意义，因为 x 与 $IP(x)$(或 y 与 $IP^{-1}(y)$)的一一对应关系是已知的。它们的作用在于打乱原来输入 x 的 ASCII 码字划分的关系，并将原来明文的校验位 $x_8, x_{16}, \cdots, x_{64}$变成 IP 的输出的一个字节。

扩展函数 E 如下：

比特选择表 E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

置换 P 如下：

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

密钥方案的计算：每一轮都使用不同的、从初始密钥(又称种子密钥)k 导出的 48-比特密钥 k_i。k 是一个长度为 64 的比特串，实际上它只有 56-比特密钥，在第 8,16,…,64 位为校验比特，共 8 个，这主要是为了检错。在位置 8,16,…,64 的比特是按下述办法给出的：使得每一个字节(8 比特长)含有奇数个 1。因此在每一个字节中的一个错误能被检测出。在密钥方案的计算中，不考虑校验比特。密钥方案的计算过程如下：

(1)给定一个 64-比特的密钥 k，删掉 8 个校验比特并利用一个固定的置换 PC-1 置换

k 的剩下的 56 比特，记 $PC-1(k)=C_0D_0$，这里 C_0 是 $PC-1(k)$的前 28 比特，D_0 是 $PC-1(k)$的后 28 比特。

(2)对每一个 i，$1\leqslant i\leqslant 16$，计算

$$C_i = LS_i(C_{i-1})$$
$$D_i = LS_i(D_{i-1})$$
$$k_i = PC-2(C_iD_i)$$

其中 LS_i 表示一个或两个位置的左循环移位，当 $i=1,2,9,16$ 时，移一个位置，当 $i=3,4,5,6,7,8,10,11,12,13,14,15$ 时，移两个位置。$PC-2$ 是另一个固定置换。密钥方案的计算过程描述在图 5.2.3 中。

8 个 S-盒

	列																
行	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	S_2
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	S_3
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	S_4
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	S_5
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	S_6
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	S_7
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	S_8
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

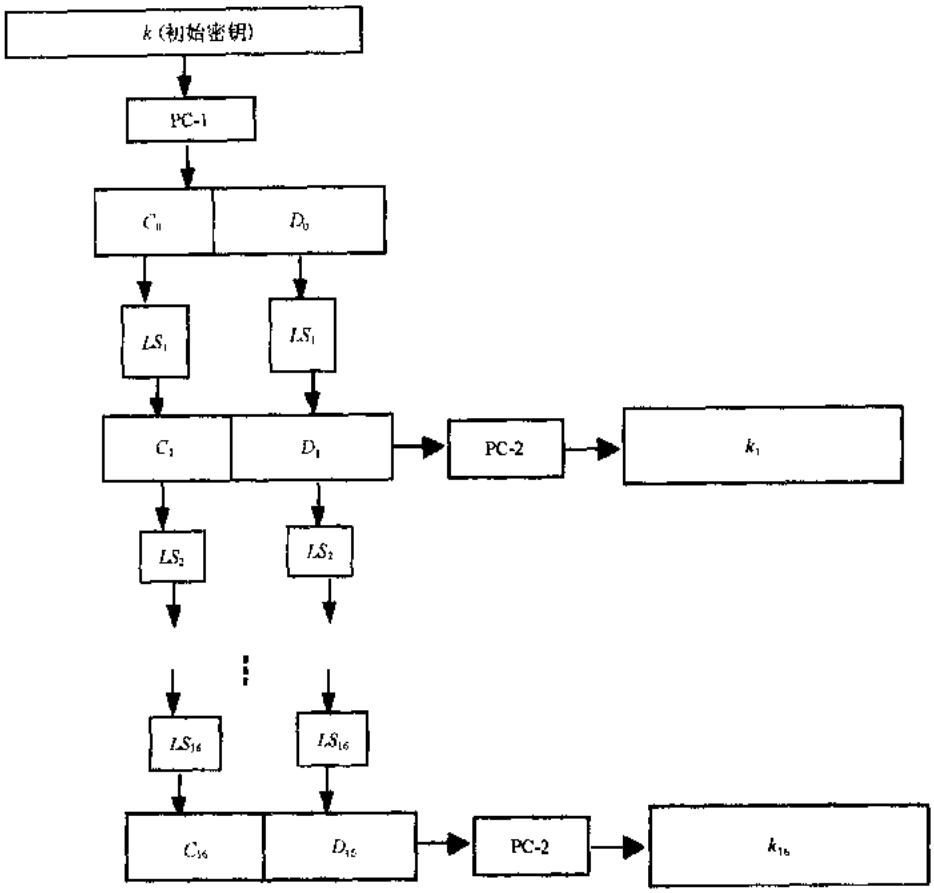


图 5.2.3 DES 的密钥方案的计算

置换 *PC*-1 和置换 *PC*-2 如下：

PC-1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	30	36	29	32

解密采用同一算法实现，把密文 y 作为输入，倒过来使用密钥方案即以逆序 k_{16}，k_{15}，…，k_1 使用密钥方案，输出将是明文 x。

5.2.2 DES 的实现

已有大量文献如[6,7,8,9,10,11,12]讨论了 DES 的硬件和软件实现方法。为了提高 DES 的软件速度，文献[13]中给出了一些很好的实用性建议。目前 DES 芯片速度的最快

记录保持者是由美国数字设备公司(DEC)开发的一个样品，数据加/解密速率达 1G 比特/秒，它能在 1 秒内加密一千五百六十万个数据分组。在 IBM3090 大型计算机上的 DES 软件实现方法每秒能完成 32000 次加密，在微机上要慢一些，但仍相当快。表 5.2.1[14,15]给出了在 Intel 和 Motorola 的几种微处理器上运算的结果和估价。

表 5.2.1 不同微处理器上的 DES 速度

处理器	速度(兆赫)	总线宽(比特)	DES 分组(每秒)
8088	4.7	8	370
68000	7.6	16	900
80286	6.0	16	1,100
68020	16.0	32	3,500
68030	16.0	32	3,900
80286	25.0	16	5,000
68030	50.0	32	9,600
68040	25.0	32	16,000
68040	40.0	32	23,200
80486	33.0	32	46,600

5.2.3 DES 的安全性

DES 的安全性完全依赖于所用的密钥。自从 DES 作为标准起，人们对它的安全性就有激烈的争论，本小节简要介绍 20 年来对 DES 的一些主要研究成果。

互补性：DES 具有性质：若明文组 x 逐位取补得 $\bar{x}$，密钥 k 逐位取补得 $\bar{k}$，且 $y=\mathrm{DES}_k(x)$，则 $\bar{y}=\mathrm{DES}_{\bar{k}}(\bar{x})$，其中 $\bar{y}$ 是 y 的逐位取补。这种特性称为算法上的互补性。这种互补性表明在选择明文攻击下仅需试验其可能的 2^{56} 个密钥的一半 2^{55} 个即可[16,17]。另外互补性告诫人们不要使用互补密钥。

弱密钥和半弱密钥：大多数密码都有某些明显的“坏”密钥，DES 也不例外[18,19]。若 $\mathrm{DES}_k(\cdot)=\mathrm{DES}_k^{-1}(\cdot)$，即如果 k 确定的加密函数与解密函数一致，则称 k 是一个弱密钥。

DES 至少有 4 个弱密钥，因为在产生密钥时，初始密钥被分成了两半，每半各自独立地移位，如果每一半的所有位都是 0 或 1，那么密钥方案中的所有密钥都是相同的，即 $k_1=k_2=\cdots=k_{16}$，这样 $\mathrm{DES}_k(\cdot)=\mathrm{DES}_k^{-1}(\cdot)$。易知，这样的情况至少有 4 种可能，很可能不存在其它弱密钥。

若存在一个不同的密钥 k' 使 $\mathrm{DES}_k(\cdot)=\mathrm{DES}_{k'}(\cdot)$，则称 k 是一个半弱密钥。此时我们也称密钥 k 和 k' 是对合的。半弱密钥的特点是成对地出现。

DES 至少有 12 个半弱密钥，因为产生 $C_0=$[1010…10]和 $D_0=$[00…0]或[11…1]或[1010…10]或[0101…01]的密钥与产生 $C_0=$[0101…01]和 $D_0=$[00…0]或[11…1]或[0101…01]或[1010…10]的密钥是互为对合的，同样地与 $C_0=$[0101…01]，$D_0=$[1010…10]或 $D_0=$[0101…01]的密钥也是互为对合的，这样就至少有 $\frac{4\times4+4\times2}{2}=12$ 个半弱密

钥，好像不存在另外的半弱密钥。

弱密钥和半弱密钥直接引起的唯一"危险"是对多重加密(在5.4节中介绍)——第二次加密使第一次加密复原。如果随机地选择密钥，则在总数2^{56}个密钥中，弱密钥和半弱密钥所占比例极小，因此弱密钥和半弱密钥的存在不会危及DES的安全性。

密文与明文、密文与密钥的相关性：文献[20]详细研究了DES的输入明文与密文以及密钥与密文之间的相关性。研究结果表明可使每个密文比特都是所有明文比特和所有密钥比特的复合函数，并且指出要达到这一要求至少需迭代5轮。文献[21]用χ^2检验证明，迭代8轮后输出和输入就可认为是不相关的了。

S-盒的设计：S-盒是DES的心脏，DES靠它实现非线性变换，关于S-盒的设计准则还没有完全公开。许多密码学家怀疑NSA设计S-盒时隐藏了"陷门"，使得只有他们才可以破译算法，但没有证据能表明这一点。在1976年，NSA披露了S-盒的下面几条设计原则：

P_0. 每一个S-盒的每一行是整数0～15的一个置换；

P_1. 每个S-盒的输出都不是它的输入的线性或仿射函数；

P_2. 改变S-盒的一个输入比特，其输出至少有2比特发生变化；

P_3. 对任何S-盒和任何输入x，$S(x)$和$S(x\oplus 001100)$至少有2比特不同(这里x是一个长度为6的比特串)；

P_4. 对任何S-盒和任何输入x，以及$e,f\in\{0,1\}$，$S(x)\neq S(x\oplus 11ef00)$，其中$x$是一个长度为6的比特串；

P_5. 对任何S-盒，当它的任一输入位保持不变，其它5位输入变化时，输出数字中的0和1的总数接近相等。

围绕S-盒人们研究了一系列相关问题，有兴趣的读者可参阅文献[22,23,24,25,26,27,28]。

密钥搜索机：对DES安全性批评意见中，较为一致的看法是DES的密钥太短，其密钥长度为56比特，密钥量为$2^{56}\approx 10^{17}$个，不能抵抗穷搜索攻击，事实证明的确如此。1997年1月28日，美国的RSA数据安全公司在RSA安全年会上公布了一项"秘密密钥挑战"竞赛，分别悬赏1000美金、5000美金和10000美金用于攻破不同密钥长度的RC5密码算法，同时还悬赏1000美金破译密钥长度为56比特的DES。RSA发起这场挑战赛是为了调查Internet上分布式计算的能力，并测试不同密钥长度的RC5算法和密钥长度为56比特的DES算法的相对强度。到目前为止，密钥长度为40比特和48比特的RC5算法已被攻破，美国克罗拉多州的程序员Verser从1997年3月13日起用了96天的时间，在Internet上数万名志愿者的协同工作下，于1997年6月17日成功地找到了DES的密钥，获得了RSA公司颁发的10000美金的奖励。这一事件表明依靠Internet的分布式计算能力，用穷搜索方法破译DES已成为可能，从而使人们认识到随着计算能力的增强，必须相应地增加算法的密钥长度。

1977年，Diffie和Hellman[29]曾建议制造每秒能测试10^6个密钥的VLSI芯片，将这样的100×10^4个芯片并行操作搜索整个密钥空间大约需1天时间。他们估计制造这样的一台机器需耗资大约2000万美元。

1993年，Wiener[30]给了一个详细的设计密钥搜索机的方案。他建议制造每秒能测试5×10^7个密钥的芯片，基于这种芯片的机器将流水作业使得16次加密同时发生。目前制

造这种芯片每片需耗资 10.50 美元，耗资 10 万美元能建造一个由 5760 个芯片组成的框架，这将使得搜索一个 DES 密钥平均大约 1.5 天。使用 10 个这样的框架建造的机器将耗资 100 万美元，搜索一个 DES 密钥平均大约 3.5 小时。

据新华社 1998 年 7 月 22 日消息，电子边境基金学会(EFF)使用一台 25 万美金的电脑在 56 小时内破译了 56 位 DES。

除了上面介绍的几个方面外，20 年来还发表了许多有关 DES 的其它方面的研究工作，这些研究不仅深入分析、检验了 DES 的各个方面，而且也大大地推动了密码学的研究与发展。

DES 的攻击方法：目前攻击 DES 的主要方法有差分攻击[31]、线性攻击[32]和相关密钥攻击[33]等方法，在这些攻击方法中，线性攻击方法是最有效的一种方法。一些有代表性的攻击方法的基本思想和观点将在 5.5 节中详细阐述。

DES 的专利权：IBM 公司拥有 DES 的专利权。

5.3 其它分组密码

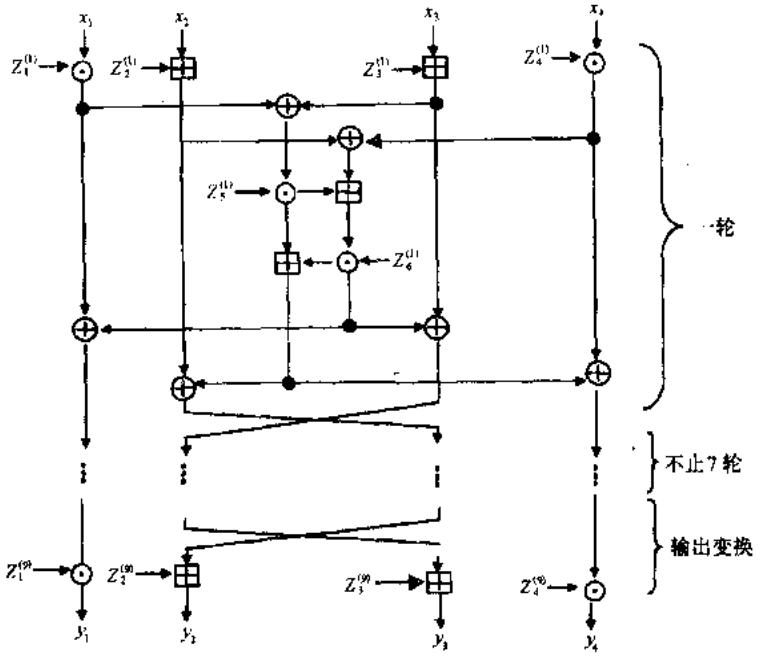
除了上节介绍的 DES 外，目前已提出了许多私钥分组密码算法，诸如 IBM 公司 70 年代早期设计的 Lucifer 算法[3,4,33]，Madryga 于 1984 年提出的 Madryga 算法[34,35]，Scott 于 1985 年提出的 New DES 算法[36]，Shimizu 等人设计的 Feal-N 算法[31,37,38,39,40,41,42]、Cusick 等人设计的 RedocII[31,43]和 RedocIII 算法[44]，Brown 等人于 1990 年设计的 Loki 算法[31,33,45,46,47,48,49]，Merkle 于 1990 年设计的 Khufa 和 Khafre 算法[31,50,51]，Rivest 设计的 RC_2 和 RC_4 算法[44](RC_4 是一个流密码)及 RC_5 算法[52,53]，Lai 等人设计的 IDEA 算法[54,55]，Deamen 等人提出的 MMB 算法[56]，NSA 开发的 Skipjack 算法[44,57]，Massey 提出的 Safer K-64 算法[58,59,60,61]以及 Davida 等人在 1979 年提出的子密钥分组密码算法[62,63]等。这么多的算法不可能逐一介绍，本节仅介绍 IDEA 算法、RC_5 算法和子密钥分组密码算法，对其它算法感兴趣的读者可参阅相关的文献。

5.3.1 IDEA

X. J. Lai 和 J. L. Massey 提出的第一版 IDEA(国际数据加密算法)于 1990 年公布，当时称为 PES(建议加密标准)。1991 年，在 Biham 和 Shamir 对其采用了差分密码分析之后，设计者为抗此种攻击，增加了他们的密码算法的强度。他们把新算法称为 IPES，即改进型建议加密标准。1992 年，设计者又将 IPES 改名为 IDEA[55]。IDEA 的明文和密文分组都是 64 比特，秘密密钥的长度是 128 比特，同一算法既可用于加密又可用于解密，该算法所依据的设计思想是“混合使用来自不同代数群中的运算”。该算法所需要的“混乱”可通过连续使用三个“不相容”的群运算于两个 16 比特子块来获得，并且该算法所选择使用的密码结构可提供必要的“扩散”。该算法的密码结构的选择也考虑了该密码算法硬件和软件实现功能。

IDEA 的描述：IDEA 是由 8 轮和随后的一个输出变换组成，图 5.3.1 所示的计算框图刻划了该密码算法的整个第一轮和输出变换。

在三个不同的群运算中，要特别注意模 $2^{16}+1$ 整数乘法运算$\odot$，这里除了将 16 比特



x_i:16 比特明文子块　　y_i:16 比特密文子块

$Z_i^{(r)}$:16 比特密钥子块　　⊕:16 比特子块的逐比特异或

⊞:16 比特整数的模 2^{16} 加　　⊙:16 比特整数的模 $2^{16}+1$ 乘(其中全零子块对应于 2^{16})

图 5.3.1　IDEA 算法加密过程的计算框图

的全零子块处理为 2^{16} 外,其余 16 比特的子块均按通常处理成一个整数的二进制表示对待,例如,$(0,0,\cdots,0)\odot(1,0,\cdots,0)=(1,0,\cdots,0,1)$,这是因为 $2^{16}\cdot 2^{15}\bmod(2^{16}+1)=2^{15}+1$。

64 比特明文块 x 被分成 4 个 16 比特子块 x_1,x_2,x_3,x_4,即 $x=x_1x_2x_3x_4$,然后这 4 个 16 比特明文子块被变成 4 个 16 比特的密文子块 y_1,y_2,y_3,y_4,即 $y=y_1y_2y_3y_4$。由明文变密文是在 52 个 16 比特的密钥子块控制下进行的,而这 52 个 16 比特密钥子块又是从 128 比特秘密密钥通过下面所描述的方式形成的。对 $r=1,2,\cdots,8$,第 r 轮要用到六个密钥子块,这六个密钥子块表示为 $Z_1^{(r)},Z_2^{(r)},\cdots,Z_6^{(r)}$。该密码算法的输出变换中用到 4 个 16 比特密钥子块,将这四个子块表示为 $Z_1^{(9)},Z_2^{(9)},Z_3^{(9)},Z_4^{(9)}$。

在每一轮中,执行的过程如下:

(1)x_1 和 $Z_1^{(1)}$ 相乘。

(2)x_2 和 $Z_2^{(1)}$ 相加。

(3)x_3 和 $Z_3^{(1)}$ 相加。

(4)x_4 和 $Z_4^{(1)}$ 相乘。

(5)将第(1)步和第(3)步的结果相异或。

(6)将第(2)步和第(4)步的结果相异或。

(7)第(5)步的结果与 $Z_5^{(1)}$ 相乘。

(8)第(6)步和第(7)步的结果相加。

(9)第(8)步的结果与 $Z_6^{(1)}$ 相乘。

(10)第(7)步和第(9)步的结果相加。

(11)第(1)步和第(9)步的结果相异或。

(12)第(3)步和第(9)步的结果相异或。

(13)第(2)步和第(10)步的结果相异或。

(14)第(4)步和第(10)步的结果相异或。

第一轮的输出是4个子块,即(11)、(12)、(13)和(14)步的结果。将中间两个块交换(最后一轮除外)后,就是下一轮的输入。

在经过八轮运算之后,有一个最后的输出变换:

(1)x_1 和 $Z_1^{(9)}$ 相乘。

(2)x_2 和 $Z_2^{(9)}$ 相加。

(3)x_3 和 $Z_3^{(9)}$ 相加。

(4)x_4 和 $Z_4^{(9)}$ 相乘。

这四步运算的结果连到一起就是所要产生的密文。

IDEA的解密过程本质上与加密过程相同,唯一不同的是解密密钥子块 $K_i^{(r)}$ 是从加密密钥子块 $Z_i^{(r)}$ 按下列方式计算出来的:

$$(K_1^{(r)},K_2^{(r)},K_3^{(r)},K_4^{(r)}) = (Z_1^{(10-r)^{-1}}, -Z_3^{(10-r)}, -Z_2^{(10-r)}, Z_4^{(10-r)^{-1}}) \quad r=2,3,\cdots,8$$

$$(K_1^{(r)},K_2^{(r)},K_3^{(r)},K_4^{(r)}) = (Z_1^{(10-r)^{-1}}, -Z_2^{(10-r)}, -Z_3^{(10-r)}, Z_4^{(10-r)^{-1}}) \quad r=1,9$$

$$(K_5^{(r)},K_6^{(r)}) = (Z_5^{(9-r)},Z_6^{(9-r)}) \qquad r=1,2,\cdots,8$$

这里 Z^{-1} 表示 Z 的模$(2^{16}+1)$的乘法逆,亦即 $Z\odot Z^{-1}=1$,$-Z$ 表示 Z 的模 2^{16} 的加法逆,亦即 $-Z\boxplus Z=0$。

IDEA的密钥方案的形成过程如下:将128比特的用户选择密钥分成8个子块,这8个密钥子块直接用来作为最前面要用的8个密钥子块,这52个密钥子块的顺序被定义为:$Z_1^{(1)},Z_2^{(1)},\cdots,Z_6^{(1)};Z_1^{(2)},Z_2^{(2)},\cdots,Z_6^{(2)};\cdots;Z_1^{(8)},Z_2^{(8)},\cdots,Z_6^{(8)};Z_1^{(9)},Z_2^{(9)},Z_3^{(9)},Z_4^{(9)}$;将128比特的用户选择密钥循环左移25位,由此产生的128比特密钥分成8个子块,它们被选作为随后的8个密钥子块;将上述第二次所产生的128比特块再循环左移25比特并将所得的128比特密钥分成8个子块,这个过程可重复到52个16比特密钥子块完全产生为止。

IDEA的分析和专利:IDEA的密钥长度是128比特,密钥搜索攻击似乎已不可行。文献[55]说明了IDEA是抵抗差分攻击的。文献[64]考察了IDEA抵抗线性攻击的能力。文献[55,56]对IDEA的密钥进行了分析,发现一些不好的密钥,但如果随机选择密钥,这些密钥不会带来危险。关于IDEA的详细讨论可参阅文献[55]。IDEA在欧洲已获专利,专利由Ascom-Tech AG拥有。

5.3.2 RC5

RC5是由RSA公司的Rivest于1994年设计的一种新型的分组密码算法[52]。RC5具有可变的字长、可变的轮数和可变的密钥长度。一个特定的RC5表示为RC5-$w/r/b$,这里 w 是以比特表示的字(word)的尺寸($w=16,32,64$),r 是轮数($0\leqslant r\leqslant 255$),$b$ 是密钥的字节(byte)长度($0\leqslant b\leqslant 255$)。RC5有两个字长($2w$ 比特)的输入和输出。RC5的一个新颖特点是使用依赖于数据的循环,它的安全性依赖于循环运算和不同运算的混合使用,它既适

合于硬件实现又适合于软件实现。

RC5 的描述：RC5 由三部分组成，即密钥扩展算法、加密算法和解密算法。这些算法使用了下列三个基本运算和他们的逆运算：

(1)模 2^w 加法运算，表示为“+”；

(2)逐比特异或运算，表示为“⊕”；

(3)字的循环左移，$x<<<y$ 表示字 x 利用 y 的循环左移，只有 y 的 $\log_2(w)$ 个低位比特用来确定 x 的循环数，所以 y 可理解为模 w。

我们首先介绍 RC5 的加密和解密算法。在加密之前，需要使用用户的秘密密钥完成一个扩展的密钥表 $S_{[0,1,\cdots,t-1]}$，$S_{[0,1,\cdots,t-1]}$的长度为 $t=2r+2$ 个字。我们先假定密钥扩展已经完成，输入分组用两个 w-比特的寄存器 A 和 B 给定，输出也放在寄存器 A 和 B 中。加密过程如下：

$$A = A + S_{[0]};$$
$$B = B + S_{[1]};$$
$$\text{For } i = 1 \text{ to } r \text{ do}$$
$$A = ((A \oplus B) <<< B) + S_{[2i]};$$
$$B = ((B \oplus A) <<< A) + S_{[2i+1]};$$

由 RC5 的加密过程知，RC5 的每一轮对寄存器 A 和 B 都进行了更新，而 DES 的每一轮只更新其中的一个。这样也可能 RC5 的半轮相当于 DES 的一轮。

解密过程与加密过程相类似，容易从加密过程中获得，只是将循环左移变为循环右移，模 2^w 加法运算变为模 2^w 减法运算。

下面我们介绍 RC5 的密钥扩展算法。密钥扩展算法由三个简单的算法构成。密钥扩展算法中使用的两个常数定义为

$$P_w = \text{odd}((e-2)2^w)$$
$$Q_w = \text{odd}((\text{phi}-2)2^w)$$

这里 $e=2.718281828459\cdots$(自然对数的基底)，$\text{phi}=1.618033988749\cdots$(黄金分割比)，$\text{odd}(x)$是不小于 x 的最小奇整数。

密钥扩展的第一个简单算法是，把 b 个字节长的秘密密钥 $K_{[0,1,\cdots,b-1]}$复制成 $c=\lceil b/\mu \rceil$个字长的一个矩阵 $L_{[0,1,\cdots,c-1]}$。以一个自然的方式完成这个操作，使用 K 的 μ 个连续的密钥字节从低阶字节到高阶字节依次填充 L 中的彼此相邻的每一个字，L 中的任何没有被填充的字节位置用零来填充。其中 $\mu=w/8$ 表示每个字中的字节数，$\lceil x \rceil$表示不小于 x 的最小整数。

密钥扩展的第二个简单算法是，利用常数 P_w 和 Q_w 获得初始化矩阵 S：

$$S_{[0]} = P_w;$$
$$\text{For } i = 1 \text{ to } t-1 \text{ do}$$
$$S_{[i]} = S_{[i-1]} + Q_w;$$

密钥扩展的第三个简单算法是，把用户的秘密密钥 K 混合到 S 之中：

$i = j = 0$;

$A = B = 0$;

执行下列步骤 $3\max(t,c)$ 次：

$A = S_{[i]} = (S_{[i]} + A + B) <<< 3$;

$B = L_{[j]} = (L_{[j]} + A + B) <<< (A + B)$;

$i = (i + 1) \bmod t$;

$j = (j + 1) \bmod c$;

密钥扩展函数有一定的"单向性"，也就是说从 S 确定 K 是不容易的。

RC5 的安全性：RC5 的一个新颖特点是使用依赖于数据的循环，也就是依赖于输入的数据来确定循环数，而不能被预先确定循环数。它的安全强度主要依赖于循环运算和不同运算的混合使用。文献[53,91]分析了 RC5 抵抗线性攻击和差分攻击的能力，分析表明 RC5-32/12/b 能抵抗线性攻击和差分攻击。事实表明，密钥长度为 40 比特和 48 比特的 RC5 不能抵抗穷搜索攻击。因为 RC5 是一个比较新的密码算法，所以它的安全性及有关问题还有待于进一步研究。

5.3.3 子密钥分组密码

Davida 等人在 1979 年引入一种新的适用于数据库的分组密码[62]。数据库被模拟为一组记录，每个记录有 t 个字段。每个记录作为一个单元进行加密，而解密时可按字段进行解密。访问一个特定的字段要求一个特定的用于此字段的子密钥。为解读每一字段采用各自的读子密钥 $d_1, d_2, \cdots, d_t$，为加密每一字段采用各自的写子密钥。对于数据库，子密钥是全局性的，即所有记录都用同样子密钥加密。给每个用户的只是允许该用户可以读或写字段的密钥。

我们先介绍一个简化的但不安全的变型方案，然后讨论为了安全性所需的修正。这个方案以中国剩余定理为基础(参见本书附录)。为了构造子密钥，每个读密钥 d_j 选择为大于字段 j 的最大可能值的随机素数。令 $n = d_1 d_2 \cdots d_t$，写子密钥为

$$e_j = (n/d_j) y_j \tag{5.3.1}$$

其中

$$e_j = (n/d_j) \cdot y_j = 1 \bmod d_j$$

令 M 是用于字段 $m_1, m_2, \cdots, m_t$ 的明文记录，整个记录被加密成

$$C = \sum_{j=1}^{t} e_j m_j \mod n \tag{5.3.2}$$

因为 C 是方程

$$x = m_j \bmod d_j \qquad j = 1, 2, \cdots, t \tag{5.3.3}$$

的解，所以第 j 个字段只用读子密钥 d_j 就易于将它解密。图 5.3.2 图示了一个记录的加密和解密的整个过程。

第 j 个字段可只用读和写子密钥 d_j 和 e_j 更新。令 m_j' 表示新值，更新的密文 C' 由

$$C' = [C - e_j(C \bmod d_j) + e_j m_j'] \bmod n \tag{5.3.4}$$

给定。

上述方案有两个弱点。第一个是在方程(5.3.2)所定义的加密公式中，令 m_{ij} 表示记录

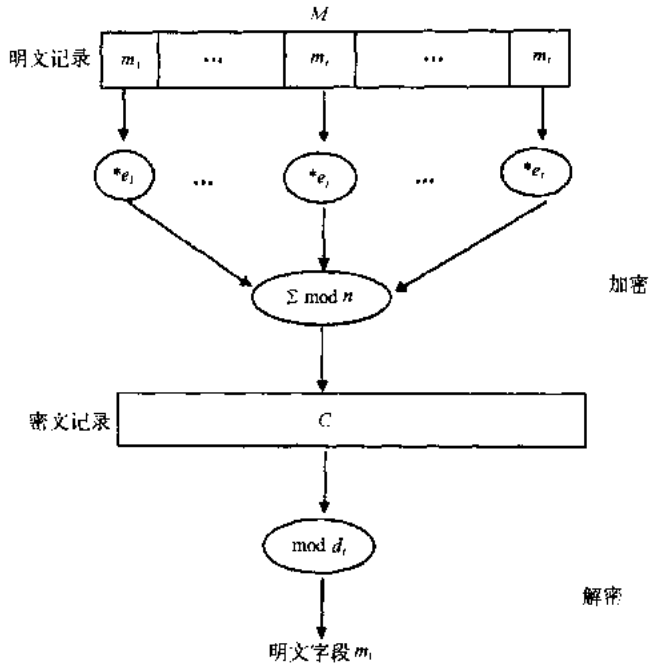


图 5.3.2　子密钥分组密码的加密和解密框图

i 中第 j 个字段的值。用户知道两个记录 M_1 和 M_2 的明文值 m_{1j}和 m_{2j}就可从密文 C_1 和 C_2 确定读密钥 d_j。由方程(5.3.3)可知，对某 μ_1 和 μ_2 有

$$C_1 - m_{1j} = \mu_1 d_j$$
$$C_2 - m_{2j} = \mu_2 d_j$$

因此，通过计算 $C_1 - m_{1j}$和 $C_2 - m_{2j}$的最大公因子就能以很高的概率决定 d_j。解决办法是在用方程(5.3.2)加密之前对每个 m_j 附加一个随机的 32 比特(或更长)的值 x_j。

第二个弱点是个别地更新字段的方法，如方程(5.3.4)所定义的，可能暴露读密钥。令 C_1 和 C_2 是两个密文记录，每个有 t 个字段，并假定每个记录中的前 $t-1$ 个字段被修改成由方程(5.3.4)定义的。令 C_1' 和 C_2' 表示修改的密文记录，这样

$$C_1 - C_1' = 0 \bmod d_t$$
$$C_2 - C_2' = 0 \bmod d_t$$

即有某 v_1 和 v_2 使得

$$C_1 - C_1' = v_1 d_t$$
$$C_2 - C_2' = v_2 d_t$$

因此，通过计算 $C_1 - C_1'$ 和 $C_2 - C_2'$ 的最大公因子就能以很高的概率决定 d_t。解决办法是对所有字段采用新的随机值 x_j，重新加密整个记录。

因为字段 j 的写子密钥 e_j 可从 n 和读子密钥 d_j 计算出，所以具有访问所有读子密钥的任何用户组，都可计算 n，并决定所有的写子密钥。为了防止串通，对每个记录附加哑字段，这些字段的读和写密钥不发给任何用户。

5.4　分组密码的工作模式

直接使用分组密码算法的工作模式称为电码本(ECB)模式，在 5.1 节我们已指出了

这种工作模式存在的缺陷，为了克服这些缺陷，我们不得不改变工作模式，当然也有别的目的和原因。所谓一个分组密码的“工作模式”就是以这个分组密码为基础用不同的方式构造一个分组密码系统。目前已提出了许多种分组密码的工作模式，诸如密码分组链接(CBC)模式、密码反馈(CFB)模式、输出反馈(OFB)模式、级连模式(CM，又称多重加密)、计数器模式、分组链接(BC)模式、扩散密码分组链(PCBC)模式等。其中 CBC 模式、CFB 模式、OFB 模式和 CM 是四种最主要和最基本的模式。本节我们以 DES 为例来介绍这四种模式，对其它模式感兴趣的读者可参阅文献[44,65]。

密码分组链接(CBC)模式：图 5.4.1 图示了 CBC 模式。在 CBC 模式下，每个明文组 x_i 加密之前，先与反馈至输入端的前一组密文 y_{i-1} 按位模 2 求和后，再送至 DES 加密，即 $y_i=\mathrm{DES}_K(x_i\oplus y_{i-1})$，其中 $y_0=\mathrm{IV}$ 是一个初始向量，无需保密，但需随消息更换，收发双方必须选用同一个 IV。显然各密文组 y_i 不仅与当前明文组 x_i 有关，而且通过反馈作用还与以前的明文组 $x_1,x_2,\cdots,x_{i-1}$ 有关。解密过程为 $x_i=\mathrm{DES}_K^{-1}(y_i)\oplus y_{i-1}$。

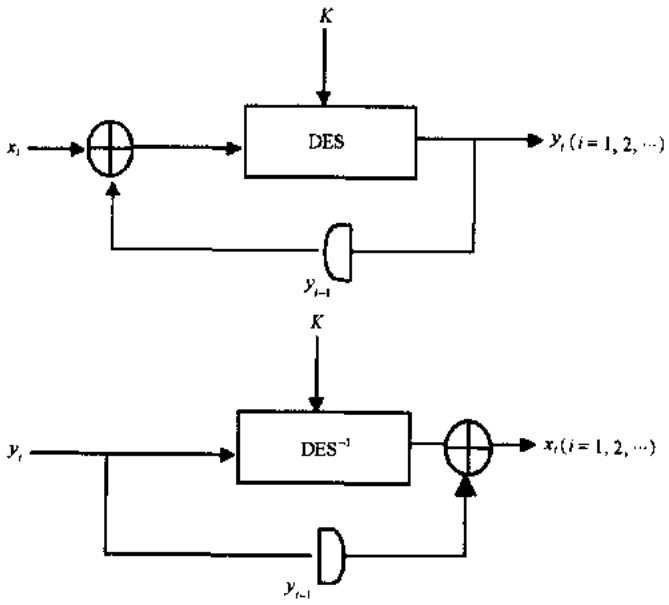


图 5.4.1 CBC 模式

CBC 模式的优点有二：一是能够隐蔽明文的数据模式，$x_{i+n}=x_i$ 未必蕴含着 $y_{i+n}=y_i$；二是在某种程度上能防止数据窜改，诸如组的重放、嵌入和删除等。CBC 模式的缺点是会出现错误传播，密文中任一位发生变化会涉及后边一些组。CBC 模式的错误传播不大，一个传输错误至多影响两个组，但对于传输中的同步差错(增加或丢失一个或多个比特)比较敏感。

密码反馈(CFB)模式：若待加密消息必须按字符(如电传电报)或按比特处理时，可采用 CFB 模式，参见图 5.4.2。x_i 和 y_i 都为 n 比特长，n 可任选，$1\leqslant n\leqslant 64$，一般取 $n=8$。$L=[64/n]$，$[x]$表示不小于 x 的最小整数。$\tilde{x}_i$ 是取自寄存器右边的 64 比特，CFB 模式实际上是一种自同步流密码(SSSC)。在这种密码模式中，DES 提供“复杂的非线性”无记忆逻辑。CFB 模式和 CBC 模式的区别在于反馈的密文不再是 64 比特，且不是直接与明文相加，而是长度为 n 比特，且反馈至密钥产生器。

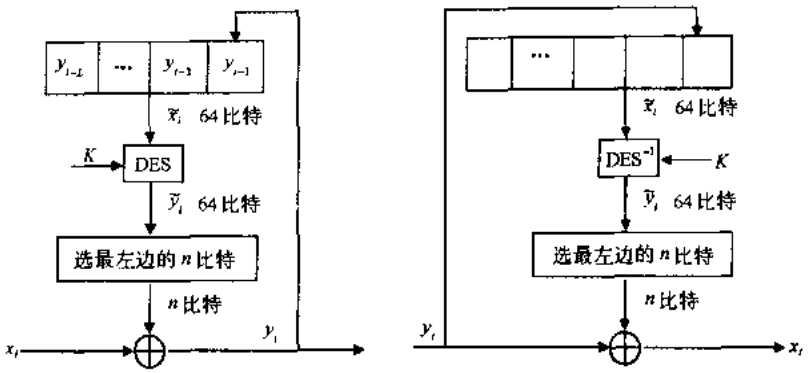


图 5.4.2 CFB 模式

CFB 模式除了具备 CBC 模式的两个优点外，还有一个优点是它特别适用于用户数据格式的需要。在密码体制设计中，应尽量避免更改已有系统的数据格式和一些规定。

CFB 模式的缺点有二：一是对信道错误较敏感，且会造成错误传播；二是数据加密的速率被降低。幸运的是，这种模式多用于数据网中较低层次，其数据速率都不太高。

CFB 模式也需要一个初始向量 IV，无需保密 IV，但对每条消息必须有一个不同的 IV。

输出反馈(OFB)模式：OFB 模式将 DES 作为一个密钥流产生器，其输出的 n-比特密钥直接反馈至 DES 的输入端，同时这 n-比特密钥和输入的 n-比特明文段进行对应位模 2 相加，参看图 5.4.3。

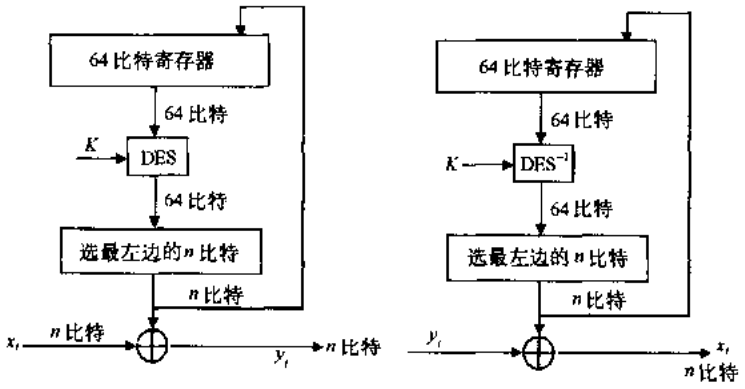


图 5.4.3 OFB 模式

OFB 模式的引入是为了克服 CBC 模式和 CFB 模式的错误传播所带来的问题。OFB 模式虽然克服了错误传播，但同时也带来了流密码的缺点。对密文被篡改难于进行检测，但由于 OFB 模式多在同步信道中运行，对手难于知道消息的起止点而使这类主动攻击不易奏效。OFB 模式不具有自同步能力，要求系统要保持严格的同步，否则难于解密。OFB 模式的初始向量 IV 无需保密，但对每条消息必须选用不同的 IV。关于这种模式的进一步讨论可参阅文献[66,67,68]。

级连模式(CM)(又称多重加密)：上述介绍的模式各有特点和用途，ECB 模式适用于对密钥加密，CFB 模式常用于对字符加密，OFB 模式常用于卫星通信中的加密。CBC 模式和 CFB 模式都可用于认证系统。但在这些模式中，密钥的长度都没有增加，为了增加密钥

的长度，有人建议将一种分组密码体制进行级联，在不同密钥作用下，连续多次对一组明文进行加密，参见图 5.4.4。

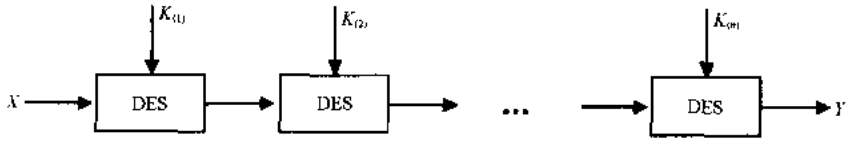


图 5.4.4 CM 或多重加密

级连模式又称多重加密，图 5.4.4 所示的情况是使用一种算法进行多重加密，当然也可以使用多种不同算法进行多重加密。关于这一模式的研究可参阅文献[67,69,70,71,72]。对付 CM 或多重加密的最有效的攻击方法是"中间相遇攻击"，"中间相遇攻击"方法的基本思想和观点将在第 7 章中介绍。在 CM 或多重加密中值得注意的另一个问题是选用的分组密码算法是否成群，如果该分组密码算法成群，那么根据群的封闭性任意次数的级连加密都等于一次加密，所以不增加破译难度。因此一个分组密码算法是否成群是一个值得考虑的问题。幸运的是，已证明 DES 不是一个群[73]，从而 DES 通过级连可提高安全性(从密钥的角度考虑)。

本节最后我们介绍一个破译级连模式(或多重加密)的一个基本原理。

定理 5.4.1[74] 破译级连密码至少与破译它的第一个子密码一样难。

证明：图 5.4.5 是我们破译级连密码所用的模型。假定密码分析者掌握一种能破译级连密码的攻击方法 A，则它可以用此破译方法 A 破译第一个子密码。破译过程如下：

(1)选择他自己的密钥 $Z_2,\cdots,Z_n$；

(2)然后将 $\tilde{y}$($\tilde{y}$ 是第一个子密码的已知密文)变换成级连密码的相应密文 y；

(3)最后应用攻击方法 A。

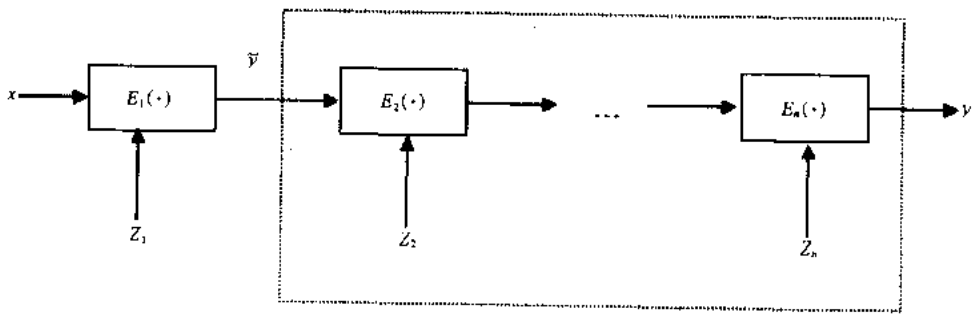


图 5.4.5 破译级连密码的模型

5.5 攻击分组密码的一些典型方法

攻击分组密码的方法很多，可能不同的分组密码有不同的攻击方法。目前比较流行和实用的方法主要有差分分析方法[31]和线性分析方法[32]。另外，时间-存贮权衡方法[92,93]也是一种比较实用的攻击方法。本节我们以 DES 为例，来介绍这三种方法的基本思想。

5.5.1 时间-存贮权衡分析方法

时间-存贮权衡(Time-memory trade-off)攻击是一种选择明文攻击。所以敌手 O 拥有明文 x 和密文 y,其中 $y=e_K(x)$,他想确定 K。时间-存贮权衡攻击的一个特点是它不依赖于 DES 的"结构",只与 DES 的输入、输出长度(64 比特)和密钥长度(56 比特)有关。

我们已经知道,破译一个密码有两种朴素的方法:穷搜索和查表法。穷搜索采用已知明文攻击,其基本观点是:给定一个明-密文对(x,y),试验所有可能的 2^{56}个密钥 K,直到 $y=e_K(x)$为止。其时间复杂度为 $T=O(2^{56})$,空间复杂度为 $S=O(1)$。查表法采用选择明文攻击,其基本观点是:对一个给定的明文 x,对所有可能的 2^{56}个密钥 K,预计算 $y=e_K(x)$。构造一张有序对表$\{(y_K,K)\}_{K\in\mathcal{K}}$,以 y_K 给出 K 的标号。因此,对于给定的密文,相应的密钥 K 在空间复杂度为 $S=O(2^{56})$时可在时间 $T=O(1)$内找出。

时间-存贮权衡攻击方法是一种混合方法,它在选择明文攻击中以时间换取空间。它比穷搜索的时间复杂度小,搜索时间仅为 $T=O(n^{2/3})$。但它比查表法的空间复杂度小,表的容量仅为 $S=O(n^{2/3})$,其中 n 为密钥量的大小。对 DES 而言,$T=O(2^{112/3})$,$S=O(2^{112/3})$。

设 R 是一个约化函数(reduction function),它将一个 64 比特长的串约化成 56 比特长的串。设 x 是一个长为 64 比特的固定的明文串,定义 $g(K_0)=R(e_{K_0}(x))$,K_0 是一个长为 56 比特的串。g 是一个从 56 比特长的串到 56 比特长的串的函数。密码分析的目的是求出 g 的逆,即 $K_0=g^{-1}(R(e_{K_0}(x)))$。

在预处理阶段,敌手 O 随机地选择 m 个长为 56 比特的串,记为 $X(i,0)(1\leqslant i\leqslant m)$。O 根据递推关系 $X(i,j)=g(X(i,j-1))(1\leqslant i\leqslant m,1\leqslant j\leqslant t)$计算 $X(i,j)(1\leqslant j\leqslant t)$,记 $X=(X(i,j))_{1\leqslant i\leqslant m,1\leqslant j\leqslant t}$,见表 5.5.1。

表 5.5.1 $X(i,j)$的计算

$$
\begin{array}{ccccccc}
X(1,0) & \xrightarrow{g} & X(1,1) & \xrightarrow{g} & \cdots & \xrightarrow{g} & X(1,t) \\
X(2,0) & \xrightarrow{g} & X(2,1) & \xrightarrow{g} & \cdots & \xrightarrow{g} & X(2,t) \\
\vdots & & & & & & \vdots \\
X(m,0) & \xrightarrow{g} & X(m,1) & \xrightarrow{g} & \cdots & \xrightarrow{g} & X(m,t)
\end{array}
$$

O 构造一张有序对表 $T=\{(X(i,t),X(i,0))\}_{1\leqslant i\leqslant m}$,以 $X(i,t)$给出 $X(i,0)$的标号。这个表所需的存贮量为 $S=O(m)$,预计算时间为 $T_p=O(mt)$。

当 O 获得一个他所选择的明文 x 的一个密文 y 时,他向前面介绍的那样先计算表 X,然后来确定密钥 K。他首先在矩阵 X 的第 t 列搜索 K,但这个也可只通过查表 T 来实现。确定办法是:O 计算 $y_1=R(y)$,看 X 的第 t 列或 T 的第 1 列中的哪一个 $X(i,t)(1\leqslant i\leqslant m)$使 $g(X(i,t))=y_1$,若能找到 $X(i,t)$使 $g(X(i,t))=y_1$,则认为找到了 K,置 $K=X(i,t)$;若找不到,则以 $t-1,t-2,\cdots,1,0$ 的次序在其余列搜索。假定 $K=X(i,t-j)$,对某一 j,$1\leqslant j\leqslant t$,也就是假定 K 是在 X 的前 t 列。于是 $g^j(K)=X(i,t)$,其中 g^j 表示将 g 迭代 j 次获得的函数。注意到 $g^j(K)=g^{j-1}(g(K))=g^{j-1}(R(e_K(x)))=g^{j-1}(R(y))$。现在我们从递推关系

$$y_j=\begin{cases}R(y) & j=1\\ g(y_{j-1}) & 2\leqslant j\leqslant t\end{cases}$$

计算 y_j。

若 $K=X(i,t-j)$，则 $y_j=X(i,t)$。但若 $y_j=X(i,t)$，则未必有 $K=X(i,t-j)$，因为约化函数 R 不是一个单射，平均地每个像有 $2^8=256$ 个原像。所以我们需要检查是否有 $y=e_{X(i,t-j)}(x)$来确定 $X(i,t-j)$是否是一个真正的密钥。我们没有存贮值 $X(i,t-j)$，但我们能很容易地从 $X(i,0)$出发，将函数 g 迭代 j 次重新计算出它。上述计算过程可描述为：

(1) 计算 $y_1=R(y)$

(2) 对 $j=1$ 到 t 完成下列步骤：

(a) 如果对某一 i，$y_j=X(i,t)$，那么从 $X(i,0)$出发，将函数 g 迭代 $t-j$ 次，计算 $X(i,t-j)$

(b) 如果 $y=e_{X(i,t-j)}(x)$，那么置 $K=X(i,t-j)$并停机

(c)计算 $y_{j+1}=g(y_j)$。

文献[92]中证明，若 $mt^2=n$，则上述算法成功的概率大约为 $0.8mt/n$。建议取 $m\approx t\approx n^{1/3}$并构造大约 $n^{1/3}$个表，每一个表使用一个不同的约化函数 R。如果这样做了，存贮量为 $O(n^{2/3})$，预计算时间为 $O(n)$，搜索时间为 $O(n^{2/3})$。关于该算法的详细分析及进一步讨论参见文献[92,93]。

5.5.2 差分分析方法

差分分析(Differential Cryptandysis)方法是一种选择明文攻击，其基本思想是通过分析特定明文差对结果密文差的影响来获得可能性最大的密钥。它主要适用于攻击迭代密码体制，迭代密码体制是以数次迭代一个“简单”的图函数为基础。诸如 DES、IDEA、Feal-N 等。虽然差分分析方法对破译 16-轮的 DES 不能提供一种实用的方法，但用它破译轮数低的 DES 是很成功的。例如，8-轮 DES 在个人计算机上几分钟就可以破译。下面我们首先来介绍差分分析的理论依据，其次给出一个简单的分析实例——对 3-轮 DES 的差分攻击。

5.5.2.1 差分分析的理论依据

因为 DES 中的初始置换 IP 及其逆置换 IP^{-1}是公开的，所以为了方便起见，我们忽略掉初始置换 IP 及其逆置换 IP^{-1}，这并不影响分析。前面我们介绍的是 16-轮 DES，实际上它可以扩展为任意轮 DES。这里只考虑 n-轮 DES，$n\leqslant 16$。在 n-轮 DES 中，我们将 L_0R_0 视作明文，L_nR_n 是密文(注意，我们也没有交换 L_n 和 R_n 的位置)。差分分析的基本观点是比较两个明文的异或与相应的两个密文的异或。一般地，我们将考虑两个具有确定的异或值 $L_0'R_0'=L_0R_0\oplus L_0^*R_0^*$ 的明文 L_0R_0 和 $L_0^*R_0^*$。

定义 5.5.1 设 S_j 是一个特定的 S-盒($1\leqslant j\leqslant 8$)，(B_j,B_j^*)是一对长度为 6 比特的串。我们说 S_j 的输入异或是 $B_j\oplus B_j^*$，S_j 的输出异或是 $S_j(B_j)\oplus S_j(B_j^*)$。

对任何 $B_j'\in Z_2^6$，记 $\Delta(B_j')=\{(B_j,B_j^*)\mid B_j\oplus B_j^*=B_j'\}$。易知，$|\Delta(B_j')|=2^6=64$，且 $\Delta(B_j')=\{(B_j,B_j\oplus B_j')\mid B_j\in Z_2^6\}$。对 $\Delta(B_j')$中的每一对，我们能计算出 S_j 的一个输出异或，共可计算出 64 个输出异或，它们分布在 $2^4=16$ 个可能的值上。将这些分布列成表。这些

分布的不均匀性将是差分攻击的基础。

例 5.5.1 设第一个 S-盒 S_1 的输入异或为 110100，那么 $\Delta(110100)=\{(000000,110100),(000001,110101),\cdots,(111111,001011)\}$。现在我们对集合 $\Delta(110100)$ 中的每一个有序对，计算 S_1 的输出异或。例如，$S_1(000000)=E_{16}=1110$，$S_1(110100)=9_{16}=1001$，所以有序对(000000,110100)的输出异或为 0111。

对 $\Delta(110100)$ 中的每一个对，都做这样的处理后，可获得下列的输出异或分布：

0000	0001	0010	0011	0100	0101	0110	0111
0	8	16	6	2	0	0	12
1000	1001	1010	1011	1100	1101	1110	1111
6	0	0	0	0	8	0	6

在例 5.5.1 中，16 个可能的输出异或中实际上只有 8 个出现。一般地，如果我们固定一个 S-盒 S_j 和一个输入异或 B_j'，那么平均地来讲，所有可能的输出异或中实际上出现大约 75%～80%。

对 $1\leqslant j\leqslant 8$，长度为 6 比特的串 B_j' 和长度为 4 比特的串 C_j'，定义

$$IN_j(B_j',C_j')=\{B_j\in Z_2^6 \mid S_j(B_j)\oplus S_j(B_j\oplus B_j')=C_j'\}$$

$$N_j(B_j',C_j')=|IN_j(B_j',C_j')|$$

$N_j(B_j',C_j')$ 表示对 S-盒 S_j 具有输入异或为 B_j'，输出异或为 C_j' 的对的数量。易知，$IN_j(B_j',C_j')$ 可分成 $N_j(B_j',C_j')/2$ 对，使得每一对的异或为 B_j'。

在例 5.5.1 中的分布表由值 $N_1(110100,C_1')$，$C_1'\in Z_2^4$ 构成。集合 $IN_1(110100,C_1')$ 中的元素如表 5.5.2 所列。

表 5.5.2 具有输入异或 110100 的所有可能输入

输出异或	可能的输入
0000	
0001	

对 8 个 S-盒中的每一个，都有 64 个可能的输入异或，所以共需计算 512 个分布。这些可通过计算机容易算出。

我们知道，在第 i 轮，S-盒的输入可写作 $B=E\oplus J$，其中 $E=E(R_{i-1})$ 是 R_{i-1} 的扩展，$J=K_i$ 由第 i 轮的密钥比特构成。此时，输入异或(对所有的 8 个 S-盒)可通过下式来计算：

$$B\oplus B^*=(E\oplus J)\oplus(E^*\oplus J)=E\oplus E^*$$

显然，输入异或不依赖于密钥比特 J。然而，输出异或必定依赖于这些密钥比特。

我们将 B,E,J 均写作长度为 6 比特的串的级联：

$$B=B_1B_2B_3B_4B_5B_6B_7B_8$$

$$E=E_1E_2E_3E_4E_5E_6E_7E_8$$

$$J=J_1J_2J_3J_4J_5J_6J_7J_8$$

并将 B^*,E^*,J^* 也写成长度为 6 比特的串的级联。让我们此时假定对某一 $j,1\leqslant j\leqslant 8$，我们知道 E_j 和 E_j^* 的值和 S_j 的输出异或的值 $C_j=S_j(B_j)\oplus S_j(B_j^*)$，则必然有 $E_j\oplus J_j\in IN_j(E_j',C_j')$，其中 $E_j'=E_j\oplus E_j^*$。

设 E_j 和 E_j^* 是两个长度为 6 比特的串，C_j' 是一个长度为 4 比特的串，定义

$$\text{test}_j(E_j,E_j^*,C_j')=\{B_j\oplus E_j\mid B_j\in IN_j(E_j',C_j')\}$$

这里

$$E_j'=E_j\oplus E_j^*$$

$\text{test}_j(E_j,E_j^*,C_j')$也就是 E_j 和集合 $IN_j(E_j',C_j')$中的每一个元素取异或所得的异或值所作成的集合。

综上所述，我们可以得出如下定理。

定理 5.5.1 设 E_j 和 E_j^* 是 S-盒 S_j 的两个输入，S_j 的输出异或是 C_j'，记 $E_j'=E_j\oplus E_j^*$，则密钥比特 J_j 出现在集合 $\text{test}_j(E_j,E_j^*,C_j')$之中，即 $J_j\in\text{test}_j(E_j,E_j^*,C_j')$。

在集合 $\text{test}_j(E_j,E_j^*,C_j')$中恰有 $N_j(E_j',C_j')$个长度为 6 比特的串，J_j 的正确值必定是这些可能值中的一个。

例 5.5.2 设 $E_1=000001,E_1^*=110101,C_1'=1101$。因为 $N_1(110100,1101)=8$，所以在集合 $\text{test}_1(000001,110101,1101)$中恰有 8 个比特串。从表 5.5.1 可知，

$IN_1(110100,1101)=\{000110,010000,010110,011100,100010,100100,101000,110010\}$

因此

$\text{test}_1(000001,110101,1101)=\{000111,010001,010111,011101,100011,100101,101001,110011\}$

如果我们有第二个这样的三重组 E_1,E_1^*,C_1'，那么我们能获得包含密钥比特 J_1 的第二个集合 test_1，则 J_1 必定是在这两个集合的交集之中。如果我们有一些这样的三重组，那么我们就能很快地确定 J_1 中的密钥比特。一个最直接的方法是：建立一个有 64 个计数器的计数矩阵，用来记录密钥比特 J_1 的 64 种可能的取值情况。每计算一个 test_1，如果某一 6 比特长的串在 test_1 之中，那么该 6 比特长的串对应的计数器增加 1，否则不增加。给定 t 个三重组 (E_1,E_1^*,C_1')，我们希望在计数矩阵中找到一个唯一的计数器，该计数器的计数值为 t，则这个计数器对应的 6 比特长的串即为密钥比特 J_1。

5.5.2.2 *差分分析的应用实例*

我们现在应用上节的观点来攻击 3-轮 DES，作为差分分析的一个应用实例。

设 L_0R_0 和 $L_0^*R_0^*$ 是两对明文，对应的密文分别为 L_3R_3 和 $L_3^*R_3^*$。我们可将 R_3 表示为

$$R_3 = L_2 \oplus f(R_2,K_3) = R_1 \oplus f(R_2,K_3) = L_0 \oplus f(R_0,K_1) \oplus f(R_2,K_3)$$

同样地

$$R_3^* = L_0^* \oplus f(R_0^*,K_1) \oplus f(R_2^*,K_3)$$

因此

$$R_3' = R_3 \oplus R_3^* = L_0' \oplus f(R_0,K_1) \oplus f(R_0^*,K_1) \oplus f(R_2,K_3) \oplus f(R_2^*,K_3)$$

其中 $L_0'=L_0 \oplus L_0^*$。

现在，假定我们选择明文使得 $R_0=R_0^*$，即 $R_0'=R_0 \oplus R_0^*=00\cdots0$（因为是选择明文攻击，所以这种假定是合理的），则

$$R_3' = L_0' \oplus f(R_2,K_3) \oplus f(R_2^*,K_3)$$

因为 L_0,L_0^*,R_3,R_3^* 为已知，所以 R_3' 和 L_0' 可计算出。这样 $f(R_2,K_3) \oplus f(R_2^*,K_3)$ 可由下式算出

$$f(R_2,K_3) \oplus f(R_2^*,K_3) = R_3' \oplus L_0'$$

又 $f(R_2,K_3)=P(C)$，$f(R_2^*,K_3)=P(C^*)$，C,C^* 分别表示 8 个 S-盒的两个输出，所以 $P(C) \oplus P(C^*)=R_3' \oplus L_0'$。而 P 是固定的、公开的、线性的，故 $C \oplus C^*=P^{-1}(R_3' \oplus L_0')$，这正是 3-轮 DES 的 8 个 S-盒的输出异或。

另外，由于 $R_2=L_3$ 和 $R_2^*=L_3^*$ 也是知道的（因为它们是密文的一部分），所以我们能使用公开知道的扩展函数 E 计算 $E=E(L_3)$ 和 $E^*=E(L_3^*)$。

对 3-轮 DES 的第 3 轮，我们已经知道 E,E^* 和 C'，现在的问题是构造 test_j，$1 \leqslant j \leqslant 8$，$J_i \in \text{test}_j$。其构造过程如下：

输入：$L_0R_0,L_0^*R_0^*,L_3R_3$ 和 $L_3^*R_3^*$，其中 $R_0=R_0^*$。

(1)计算 $C'=P^{-1}(R_3' \oplus L_0')$

(2)计算 $E=E(L_3)$ 和 $E^*=E(L_3^*)$

(3)对 $j=1,2,\cdots,8$，计算 $\text{test}_j(E_j,E_j^*,C_j')$。

我们通过建立 8 个具有 64 个计数器的计数矩阵，最终只能确定 K_3 中的 $6\times8=48$ 比特密钥，而其余的 $56-48=8$ 比特可通过搜索 $2^8=256$ 种可能的情况来确定。

下面我们用一个实例来图示 3-轮 DES 的攻击过程。

例 5.5.3 假定我们有下列的三对明文和密文，这里明文具有确定的异或，并且使用同一个密钥加密。为简单起见，我们使用 16 进制表示。

我们从第 1 对计算第 3 轮 S-盒的输入，它们分别是

$$E=E(L_3)=000000000111111000001110100000000110100000001100$$

$$E^*=E(L_3^*)=101111110000001010101100000001010100000001010010$$

S-盒的输出异或是

明文	密文
748502CD38451097 3874756438451097	03C70306D8A09F10 78560A0960E6D4CB
486911026ACDFF31 375BD31F6ACDFF31	45FA285BE5ADC730 134F7915AC253457
357418DA013FEC86 12549847013FEC86	D8A31B2F28BBC5CF 0F317AC2B23CB944

$$C'=C\oplus C^*=P^{-1}(R_3'\oplus L_0')=10010110010111010101101101100111$$

从第 2 对计算第 3 轮 S-盒的输入，它们分别是

$$E=101000001011111111110100000101010000001011110110$$

$$E^*=100010100110101001011110101111110010100010101010$$

S-盒的输出异或是

$$C'=10011100100111000001111101010110$$

从第 3 对计算第 3 轮 S-盒的输入，它们分别是

$$E=111011110001010100000110100011110110100101011111$$

$$E^*=000001011110100110100010101111110101011000000100$$

S-盒的输出异或是

$$C'=11010101011101011101100100101011$$

现在，我们建立 8 个具有 64 个计数器的计数矩阵。将这三对中的每一对都进行计数。我们现在图示第 1 对关于 J_1 的计数矩阵的计数过程。

在第 1 对中，我们有

$E_1'=101111, C_1'=1001, IN_1(101111,1001)=\{000000,000111,101000,101111\}$

因为 $E_1=000000$，所以我们有

$$J_1\in \text{test}_1(000000,101111,1001)=\{00000,000111,101000,101111\}$$

因此，我们在 J_1 的计数矩阵中的位置 0,7,40 和 47 处增加 1。

这里我们将一个长度为 6 比特的串视作一个 0～63 之间的整数的二元表示，用 64 个值对应位置 0,1,2,…,63。最终的计数矩阵如下：

J_1															
1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

J_2															
0	0	0	1	0	3	0	0	1	0	0	1	0	0	0	0
0	1	0	0	0	2	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0
0	0	1	1	0	0	0	0	1	0	1	0	2	0	0	0

J_3															
0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1
0	2	0	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0

J_4															
3	1	0	0	0	0	0	0	0	0	2	2	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	1
1	1	1	0	1	0	0	0	0	1	1	1	0	0	1	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	2	1

J_5															
0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
0	0	0	0	2	0	0	0	3	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	2	0	0	0	0	0	0	1	0	0	0	0	2	0

J_6															
1	0	0	1	1	0	0	3	0	0	0	0	1	0	0	1
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0

J_7															
0	0	2	1	0	1	0	3	0	0	0	1	1	0	0	0
0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	2	0	0	0	2	0	0	0	0	1	2	1	1	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1

J_8															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1
0	3	0	0	0	0	1	0	0	0	0	0	0	0	0	0

在 8 个计数矩阵的每一个中，都有唯一的一个计数器具有值 3。这些计数器的位置确定 $J_1, J_2, \cdots, J_8$ 中的密钥比特。这些位置分别是 47,5,19,0,24,7,7,49。将这些整数转化为二进制数，我们可获得 $J_1, J_2, \cdots, J_8$：

$J_1 = 101111$

$J_2 = 000101$

$J_3 = 010011$

$J_4 = 000000$

$J_5 = 011000$

$J_6 = 000111$

$J_7 = 000111$

$J_8 = 110001$

我们现在可通过查找第 3 轮的密钥方案能构造出密钥的 48 比特。密钥 K 具有下列形式：

0 0 0 1 1 0 1　0 1 1 0 0 0 1　0 1 ? 0 1 ? 0　1 ? 0 0 1 0 0

0 1 0 1 0 0 1　0 0 0 0 ? ? 0　1 1 1 ? 1 1 ?　? 1 0 0 0 1 1

这里已略去了校验比特，"?"表示一个未知的密钥比特。完全的密钥是(用 16 进制表示，并包括校验比特)1A624C89520DEC46。

5.5.3 线性分析方法

线性分析(linear cryptanalysis)方法本质上是一种已知明文攻击方法. 这种方法可用 2^{21}个已知明文破译 8-轮 DES，可用 2^{47}个已知明文破译 16-轮 DES. 该种方法在某些情况下，可用于唯密文攻击，详细讨论参阅文献[32]。线性分析的基本思想是通过寻找一个给定密码算法的有效的线性近似表达式来降低密钥的熵，从而破译密码系统。

为讨论方便起见，一方面我们忽略了 DES 的初始置换 IP 及其逆置换 IP^{-1}，因为 IP 及 IP^{-1}是公开的，所以这样做不会影响攻击；另一方面我们引入如下记号：

P：表示 64 比特的明文；

C：表示相应的 64 比特的密文；

P_H：表示 P 的左边 32 比特；

P_L：表示 P 的右边 32 比特；

C_H：表示 C 的左边 32 比特；

C_L：表示 C 的右边 32 比特；

X_i：表示第 i 轮的 32 比特中间值；

K_i：表示第 i 轮的 48 比特子密钥；

$f(X,K)$：表示 *DES* 的 f 函数；

$A[i]$：A 的第 i 比特；

$A_{[i,j,\cdots,k]}$：表示 $A_{[i]}\oplus A_{[j]}\oplus\cdots\oplus A_{[k]}$。

5.5.3.1 线性分析的基本原理

线性密码分析的目的是寻找一个给定密码算法的下列形式的"有效的"线性表达式：

$$P_{[i_1,i_2,\cdots,i_a]}\oplus C_{[j_1,j_2,\cdots,j_b]}=K_{[k_1,k_2,\cdots,k_c]} \tag{5.5.1}$$

这里 $i_1,i_2,\cdots,i_a,j_1,j_2,\cdots,j_b$ 和 $k_1,k_2,\cdots,k_c$ 表示固定的比特位置，并且对随机给定的明文 P 和相应的密文 C(K 是被使用的密钥)，等式(5.5.1)成立的概率 $p\neq\frac{1}{2}$. 用 $\left|p-\frac{1}{2}\right|$ 来表示等式(5.5.1)的有效性。

一旦我们成功地获得一个有效的线性表达式，通过下列的基于最大似然方法的算法确定一个密钥比特 $K_{[k_1,k_2,\cdots,k_c]}$是可能的：

算法 1

第 1 步：设 T 是使得等式(5.5.1)的左边等于 0 的明文的数目.

第 2 步：如果 $T>N/2$(N 表示明文的数)，那么当 $p>\frac{1}{2}$时，猜定 $K_{[k_1,k_2,\cdots,k_c]}=0$；当 $p<1/2$ 时，猜定 $K_{[k_1,k_2,\cdots,k_c]}=1$。否则，当 $p>\frac{1}{2}$时，猜定 $K_{[k_1,k_2,\cdots,k_c]}=1$；当 $p<\frac{1}{2}$时，猜定 $K_{[k_1,k_2,\cdots,k_c]}=0$。

文献[32]中指出，当 $\left|p-\frac{1}{2}\right|$ 充分小时，算法 1 成功的概率是

$$\frac{1}{\sqrt{2\pi}}\int_{-2\sqrt{N}|p-1/2|}^{\infty}e^{-x^2/2}dx$$

这个成功率只依赖于 $\sqrt{N}\left|p-\frac{1}{2}\right|$，并随着 N 或 $\left|p-\frac{1}{2}\right|$ 的增加而增加. 我们把最有效的线性表达式(也就是 $\left|p-\frac{1}{2}\right|$ 是最大的)称作最佳逼近式，相应的概率 p 称作最佳概率。

式(5.5.1)成立的概率可用下列的堆积引理(piling-up lemma)来计算：

引理 5.5.1 (piling-up lemma) 设 $X_i(1\leqslant i\leqslant n)$是独立的随机变量，$p(X_i=1)=p_i$，$p(X_i=0)=1-p_i$，则

$$p(X_1\oplus X_2\oplus\cdots\oplus X_n=0)=1/2+2^{n-1}\prod_{i=1}^{n}(p_i-1/2) \tag{5.5.2}$$

引理 5.5.1 可通过对 n 作归纳法来证明.

在实际中对 n-轮 DES 密码进行已知明文攻击时，我们使用(n-1)-轮 DES 的最佳表达式. 也就是说，我们假定已把最后一轮使用 K_n 作了解密，解密结果的左边 32 比特为 $C_H\oplus f(C_L,K_n)$，右边 32 比特为 C_L. 这时，我们把$(C_H\oplus f(C_L,K_n))C_L$ 当作(n-1)-轮 DES 的密文，使用(n-1)-轮 DES 的最佳表达式可获得

$$P_{[i_1,i_2,\cdots,i_a]}\oplus C_{[j_1,j_2,\cdots,j_b]}\oplus f(C_L,K_n)[e_1,e_2,\cdots,e_d]=K_{[k_1,k_2,\cdots,k_c]} \tag{5.5.3}$$

表达式(5.5.3)与 f 函数及 K_n 有关。如果在等式(5.5.3)中代入一个不正确的候选者 K_n，那么这个等式的有效性显然就降低了。因此，可使用最大似然方法来推导 K_n 和

$K_{[k_1,k_2,\cdots,k_c]}$，其算法如下：

算法 2

第 1 步：对 K_n 的每一个候选者 $K_n^{(i)}(i=1,2,\cdots)$，设 T_i 是使得等式(5.5.3)的左边等于 0 的明文的数目。

第 2 步：设 $T_{\max}=\max\{T_i\}_{i\geqslant 1}$，$T_{\min}=\min\{T_i\}_{i\geqslant 1}$。

如果 $|T_{\max}-N/2|>|T_{\min}-N/2|$，那么将 $T_{\max}$ 所对应的密钥候选者作为 K_n 并猜定 $K_{[k_1,k_2,\cdots,k_c]}=0$(当 $p>\frac{1}{2}$ 时)或 1(当 $p<\frac{1}{2}$ 时)。

如果 $|T_{\max}-N/2|<|T_{\min}-N/2|$，那么将 $T_{\min}$ 所对应的密钥候选者作为 K_n 并猜定 $K_{[k_1,k_2,\cdots,k_c]}=1$(当 $p>\frac{1}{2}$ 时)或 0(当 $p<\frac{1}{2}$ 时)。其中 N 是给定的随机明文的数目。

设 p 是使得等式(5.5.3)成立的概率，对一个子密钥候选者 $K_n^{(i)}$ 和一个随机变量 X，设 $q^{(i)}$ 是使得下列等式成立的概率：

$$f(X,K_n)[e_1,e_2,\cdots,e_d]=f(X,K_n^{(i)})[e_1,e_2,\cdots,e_d] \tag{5.5.4}$$

文献[32]中指出，当 $\left|p-\frac{1}{2}\right|$ 充分小时，如果 $q^{(i)}(i=1,2,\cdots)$ 是相互独立的，那么算法 2 的成功率为

$$\int_{x=-2\sqrt{N}\,|p-1/2|}^{\infty}\left\{\prod_{K_n^{(i)}\neq K_n}\int_{-x-4\sqrt{N}(p-1/2)q^{(i)}}^{x+4\sqrt{N}(p-1/2)(1-q^{(i)})}\frac{1}{\sqrt{2\pi}}e^{-y^2/2}dy\right\}\frac{1}{\sqrt{2\pi}}e^{-x^2/2}dx \tag{5.5.5}$$

这里的积是除了 K_n 外取遍所有的子密钥候选者. 由式(5.5.5)可知，算法 2 的成功率只依赖 $e_1,e_1,\cdots,e_d$ 和 $\sqrt{N}\,|p-1/2|$. 虽然一般地 $q^{(i)}(i=1,2,\cdots)$ 不是相互独立的，但这个结果也是很实用的，它给出了成功率的一个比较好的估计。

5.5.3.2 DES 的线性分析

我们知道，S-盒是 DES 的核心部分，所以我们首先来分析 S-盒的线性逼近.

对一个给定的 S-盒 $S_i(1\leqslant i\leqslant 8)$，$1\leqslant\alpha\leqslant 63$ 和 $1\leqslant\beta\leqslant 15$，定义

$$NS_i(\alpha,\beta)=\left|\left\{x\,|\,0\leqslant x\leqslant 63,\sum_{S=0}^{5}X_{[S]}\cdot\alpha_{[S]}=\sum_{t=0}^{3}S_i(x)[t]\cdot\beta_{[t]}\right\}\right|$$

这里 $X_{[S]}$ 表示 X 的二进制表示的第 S 个比特，$S_i(x)[t]$ 表示 $S_i(x)$ 的二进制表示的第 t 个比特，$\sum$ 表示逐比特异或和，$\cdot$ 表示逐比特与运算. NS_i 度量了 S-盒 S_i 的非线性程度。对线性逼近式

$$\sum_{S=0}^{5}(X_{[S]}\cdot\alpha_{[S]})=\sum_{t=0}^{3}(S_i(x)[t]\cdot\beta_{[t]}) \tag{5.5.6}$$

而言，$p=\frac{NS_i(\alpha,\beta)}{64}$. 当 $NS_i(\alpha,\beta)\neq 32$ 时，式(5.5.6)就是一个有效的线性表达式。这时，我们也称 S_i 的输入和输出比特是相关的. 例如，$NS_5(16,15)=12$，这表明 S_5 的第 4 个输入比特和所有输出比特的异或值符合的概率为 12/64=0.19. 因此，通过考虑 f 函数中的 E 扩展和 P 置换，我们可以推出，对一个固定的密钥 K 和一个随机给定的中间输入 X，下列等式成立的概率为 0.19：

$$X_{[15]}\oplus F(X,K)[7,18,24,29]=K_{[22]} \tag{5.5.7}$$

表 5.5.3 描述了 S-盒 S_5 的分布表的一部分，这里垂直轴和水平轴分别表示 α 和 β，每个元素表示 $NS_5(\alpha,\beta)-32$. 通过计算所有的表，我们可以看出等式(5.5.6)是所有 S-盒中，最有效的线性逼近(也就是 $|NS_i(\alpha,\beta)-32|$ 是最大的)，因此，等式(5.5.7)是 f-函数的最佳逼近。

表 5.5.3 S_5 的分布表(部分)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	4	−2	2	−2	2	−4	0	4	0	2	−2	2	−2	0	−4
3	0	−2	6	−2	−2	4	−4	0	0	−2	6	−2	−2	4	−4
4	2	−2	0	0	2	−2	0	0	2	2	4	−4	−2	−2	0
5	2	2	−4	0	10	−6	−4	0	2	−10	0	4	−2	2	4
6	−2	−4	−6	−2	−4	2	0	0	−2	0	−2	−6	−8	2	0
7	2	0	2	−2	8	6	0	−4	6	0	−6	−2	0	−6	−4
8	0	2	6	0	0	−2	−6	−2	2	4	−12	2	6	−4	4
9	−4	6	−2	0	−4	−6	−6	6	−2	0	−4	2	−6	−8	−4
10	4	0	0	−2	−6	2	2	2	2	−2	2	4	−4	−4	0
11	4	4	4	6	2	−2	−2	−2	−2	−2	2	0	−8	−4	0
12	2	0	−2	0	2	4	10	−2	4	−2	−8	−2	4	−6	−4
13	6	0	2	0	−2	4	−10	−2	0	−2	4	−2	8	−6	0
14	−2	−2	0	−2	4	0	2	−2	0	4	2	−4	6	−2	−4
15	−2	−2	8	6	4	0	2	2	4	8	−2	8	−6	2	0
16	2	−2	0	0	−2	−6	−8	0	−2	−2	−4	0	2	10	−20
17	2	−2	0	4	2	−2	−4	4	2	2	0	−8	−6	2	4
18	−2	0	−2	2	−4	−2	−8	4	6	4	6	−2	4	−6	0
19	−6	0	2	−2	4	2	0	4	−6	4	2	−6	4	−2	0
20	4	−4	0	0	0	0	0	−4	−4	4	4	0	4	−4	0
21	4	0	−4	−4	4	−8	−8	0	0	−4	4	8	4	0	4
22	0	6	6	2	−2	4	0	4	0	6	2	2	2	0	0
23	4	−6	−2	6	−2	−4	4	4	−4	−6	2	−2	2	0	4
24	6	0	2	4	−10	−4	2	2	0	−2	0	2	4	−2	−4
25	2	4	−6	0	−2	4	−2	6	8	6	4	10	0	2	−4
26	2	2	−8	−2	4	0	2	−2	0	4	2	0	−2	−2	0
27	2	6	−4	−6	0	0	2	6	8	0	−2	−4	−6	−2	0
28	0	−2	2	4	0	−6	2	−2	6	−4	0	2	−2	0	0
29	4	−2	6	−8	0	−2	2	10	−2	−8	−8	2	2	0	4
30	−4	−8	0	−2	−2	−2	2	−2	2	−2	6	4	4	4	0
31	−4	8	−8	2	−6	−6	−2	−2	2	−2	−2	−8	0	0	−4
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

易知，$NS_i(\alpha,\beta)$是偶数。另外，通过计算可知，如果 $\alpha=1,32$ 或 33，那么对所有的 S_i 和 β，都有 $NS_i(\alpha,\beta)=32$。

现在，我们将 f 函数的线性逼近扩展到整个 DES 算法上。我们先以 3-轮 DES 为例。将等式(5.5.7)应用于第 1 轮，我们可得到下列的等式，该等式成立的概率为 12/64：

$$X_2[7,18,24,29]\oplus P_H[7,18,24,29]\oplus P_L[15]=K_1[22] \qquad (5.5.8)$$

同样地，将等式(5.5.7)应用于最后一轮即第 3 轮可得下列的等式，该等式成立的概率为 12/64：

$$X_3[7.18,24,29] \oplus C_H[7,18,24,29] \oplus C_L[15] = K_3[22] \tag{5.5.9}$$

将上述两式异或，我们可获得下列的 3-轮 DES 的线性逼近表达式：

$$P_H[7,18,24,29] \oplus C_H[7,18,24,29] \oplus P_L[15] \oplus C_L[15] = K_1[22] \oplus K_3[22] \tag{5.5.10}$$

对给定的随机明文 P 和相应密文 C，等式(5.5.10)成立的概率为$(12/64)^2+(1-12/64)^2=0.70$。因为等式(5.5.7)是 f 函数的最佳线性逼近，所以等式(5.5.10)是 3-轮 DES 的最佳表达式。现在我们可用算法 1 求解等式(5.5.10)来获得 $K_1[22] \oplus K_3[22]$。

通过对一系列的明密文对的分析后，可获得关于 $K_1[22]$，$K_3[22]$的一系列方程，从而可确定出 $K_1[22]$和 $K_3[22]$。

对 5-轮 DES，类似于前面的讨论，我们首先使用 $NS_1(27,4)=22$ 获得一个 f 函数的线性逼近表达式，然后将这一表达式应用于 5-轮 DES 的第 1 轮和最后一轮可得如下线性等式：

$$X[27,28,30,31] \oplus f(X,K)[15] = K[42,43,45,46] \tag{5.5.11}$$

再将等式(5.5.7)应用于 5-轮 DES 的第 2 轮和第 4 轮，并结合等式(5.5.11)可得 5-轮 DES 的一个线性逼近表达式：

$$P_H[15] \oplus P_L[7,18,24,27,28,29,30,31] \oplus C_H[15] \oplus C_L[7,18,24,27,28,29,30,31] = K_1[42,43,45,46] \oplus K_2[22] \oplus K_4[22] \oplus K_5[42,43,45,46] \tag{5.5.12}$$

等式(5.5.12)成立的概率可由引理 5.5.1 计算出，这个概率为

$$1/2 + 2^3(-10/64)^2(-20/64)^2 = 0.519$$

关于线性密码分析就谈这些，感兴趣的读者参阅文献[32]。

5.6 注记和文献

本章主要介绍了私钥分组密码算法的基本概念、设计原则和工作模式以及现有的一些有代表性的私钥分组密码算法诸如 DES、IDEA、RC5 等。

有关分组密码算法的设计原则本章主要介绍了两个方面：一个是针对安全性的 Shannon 混乱和扩散原则；另一个是针对软硬件实现的设计原则。在具体设计一个分组密码算法时，只考虑这些方面往往还不够，需要考虑别的因素诸如所使用的环境、分组长度、密钥长度、不动点、是否能抵抗住现有的一些攻击方法等诸多因素。

有关 DES 的研究成果很多，文献[76]对此作了比较好的总结。目前关于 DES 的主要攻击方法有差分攻击方法、线性攻击方法和相关密钥攻击方法，其中线性攻击方法是最有效的。文献[77,90]中研究了线性攻击和差分攻击的关系。针对差分攻击，人们研究了一系列的问题，主要研究成果参见文献[78,79,80,81,82]。针对线性攻击，人们也作了一些讨论，这些讨论可在文献[83,84,85]中找到。DES 的公布对密码学的研究与发展起了极大的推动作用，它为人们设计分组密码算法提供了极好的思路和框架。从 DES 本身的研究中人们延伸出了许多研究问题，诸如函数的扩散特性和雪崩特性、非线性性等。

目前已提出了许多私钥分组密码算法，我们不可能逐一介绍，在本章中只介绍了四个分组密码算法，即 DES、IDEA、RC5 和子密钥分组密码。对其它分组密码算法感兴趣的读者可参阅相关的文献，在 5.3 节中我们已标出了大部分已有算法的出处。

如何加强一个分组密码算法的安全性是一个非常重要而又有价值的问题。本章以DES为例介绍了一个分组密码算法的四种工作模式，特别是级连模式或称多重加密尤为人们所关注，人们希望通过级连密码来加强安全性，但对这一技术，密码学家们持有不同的意见。在级连模式中有两个问题需着重考虑：一个是所使用的分组密码是否成群的问题，已证明DES不成群；另一个是级连密码的安全性问题，Maurer和Massey对这个问题作了部分回答[74]。

最近，美国国家标准技术研究所(NIST)公布了15个AES候选算法，这些算法的基本设计思想参见附录。

参 考 文 献

[1] Shannon, C. E., Communication Theory of Secrecy System, Bell Syst. Tech. J., Vol. 28, pp. 656—715, 1949.

[2] NBS, Data Encryption Standard, FIPS PUB 46, National Bureau of Standards, Washington, D. C. (Jan. 1977).

[3] Feistel, H., Cryptography and Computer Privacy, Scientific American, Vol. 228, No. 5, pp. 15—23, 1973.

[4] Sorkin, A., Lucifer: A Cryptographic Algorithm, Cryptologia, Vol. 8, pp. 22—41, 1984.

[5] Katzan, H. Jr., The Standard data encryption Algorithm, Petrocell: Books Inc. 1977.

[6] MacMillan, D., Single Chip Encrypts Data at 14Mb/s, Electronics, Vol. 54, No. 12, pp. 161—165, 1981.

[7] Banerjee, S. K., High Speed Implementation of DES, Computers and Security, Vol. 1, pp. 261—267, 1982.

[8] Fairfield, R. C., A. Matusevich, and J. Plany, An LSI Digital Encryption Processor (DEP), Advances in Cryptology-Crypto'84, Springer-Verlag, pp. 115—143, 1985.

[9] David, M., Desmedt, Y., Goubert, J., Hoornaert F., and Quisquater, J. J., Efficient Hardware and Software Implementation of the DES, Adavces in Cryptology-Crypto'84, Springer-Verlag, pp. 144—146, 1985.

[10] Hoornaert, F., Goubert J., and Desmedt, Y., Efficient Hardware Implementation of the DES, Advances in Cryptology-Crypto'84, Springer-Verlag, pp. 147—173, 1985.

[11] Verbauwhede, I., Hoornaert, F., Vaenderwalle, J., H. De Man and R. Govaerts, Security Considerations in the Design and Implementation of a New DES Chip, Advances in Cryptology-Eurocrypt'87, Springer-Verlag, pp. 287—300, 1988.

[12] Broscius, A. G., and Smith, J. M., Exploiting Parallelism in Hardware Implementation of the DES, Advances in Cryptology-Crypto'91, Springer-Verlag, pp. 367—376, 1992.

[13] Bishop, M., An Application for a Fast Data Encryption Standard Implementation, Computing Systems, Vol. 1, No. 3, pp. 221—254, 1988.

[14] Garon, G. and Outerbridge, R., DES Watch: An Examination of the Sufficiency of the Sata Encryption Standard for Financial Institution Information Sequrity in the 1990's, Cryptologia, Vol. 15, No. 3, pp. 177—193, 1991.

[15] Jueneman, R. R. Matyas, S. M. and Meyor, C. H., Message Authentication, IEEE Communications Magazine, Vol. 23, No. 9, pp. 29—40, 1985.

[16] Meyer, C. H., and Matyas, S. M., Cryptography: A New Dimonsion in Cmputer Data Sequrity, New York, John Wiley & Sons, 1982.

[17] Chaum, D., and Evertse, J. H., Cryptanalysis of DES with a Reduced Number of Round, Advances in Cryptology-Crypto'85, Springer-Verlag, pp. 192—211, 1986.

[18] Davies, D. W., Some Regular Properties of the DES, Advances in Cryptology-Crypto'82, Plenum Pless, pp. 89—96, 1983.

[19] Moore, J. H., and Simmons, G. J., Cycle Structure of the DES with Weak and Semi-weak Keys, Advances in Cryptology-crypto'86, Springer-Verla, pp. 3—32, 1987.

[20] Meyer, C. H., Ciphertext/Plaintext and Ciphertext/Key Depondence vs. Number of Rounds for the Data Encryp

tion Standard, AFIPS Conference Proceedings-1978 NCC Vol. 47, pp. 1110—1126, 1978.

[21] Konheim, A. G., Cryptography: A Primer, John Wiley & Sons, New York, 1981.

[22] Brickell, E. F., Moore, J. H., and Purtill, M. R., Structure in the S-Box of the DES, Advances in Cryptology-Crypto'86, Springer-Verlag, pp. 3—8, 1987.

[23] Davies, D. W. and Price, W. L., Security for Computer Networks: An Introduction to Data security in Teleprocessing and Electronic Funds transfer, John Wiley & Sons, 1984.

[24] Shamir, A., On the Security of DES, Advances in Cryptology-Crypto'85, Springer-Verlag, pp. 280—281, 1986.

[25] Webster, A. F. and Traraes, S. E., On the Design of S-Box, Advances in Cryptology-Crypto'85, Springer-Verlag, pp. 523—524, 1986.

[26] Forri, R., The Strict Avalanche Criterion: Spectral Properties of Boolean Functions and an Extended Definition, Advances in Cryptology-Crypto'88, Springer-Verlag, pp. 450—468, 1990.

[27] Preneel, B. Vanlcekwijk, W., Linden, L. Van, Govaerts R. and Vandewalle, J., Propagation Characteristics of Boolean Functions, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, pp. 161—173, 1991.

[28] Gordon, J. A. and Retkin, H., Are Big S-boxes Best? IEEE Workshop on Comms. Security, Santa Barbara, 1981.

[29] Diffie, W. and Hellman, M. E., Exhaustive Cryptanalysis of the NBS Data Encryption Standard, Computer, Vol. 10, No. 6, pp. 74—84, 1977.

[30] Wiener, M. J. Efficient DES KEY SEARCH, Advances in Cryptology-Crypto'93, Springer-Verlag, Rump Session, 1994.

[31] Biham, E. and Shamir, A., Differential Cryptanalysis of the Data Encryption Standard, Springer-Verlag, 1993.

[32] Matsui, M., Linear Cryptanalysis Method for DES Cipher, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, pp. 386—397, 1994.

[33] Biham, E., New Types of Cryptanalytic Attacks using related keys, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, pp. 398—409, 1994.

[34] Madryga, W. E., A High Performance Encryption Algorithm, Computer Security: A Global Challenge, North Holland: Elsevier Science Publishers, 1984, pp. 557—570.

[35] Gustafson, H., Dawson E. and Caelli, B., Comparison of Block Ciphers, Advances in Cryptology-Auscrypt'90, Springer-Verlag, pp. 208—220, 1990.

[36] Scott, R., Wide Open Encryption Design Offers Flexible Implementations, Cryptologia, Vol. 9, No. 1, pp. 75—90, 1985.

[37] Shimizu, A. and Miyaguchi, S., Fast data Encipherment Algorighm FEAL, Advances in Cryptology-Eurocrypt'87, Springer-Verlag, pp. 267—280, 1988.

[38] Boer, B. Den., Cryptanalysis of FEAL, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, pp. 293—300, 1988.

[39] Murphy S., The Cryptanalysis of FEAL-4 with 20 Chosen Plaintexts, Journal of Cryptology, Vol. 2, No. 3, pp. 145—154, 1990.

[40] Gilbert H. and Chasse, G., A Statistical Attack on the FEAL-8 Cryptosystem, Advances in Cryptology-Crypto'90, Springer-Verlag, pp. 22—33, 1990.

[41] Matsui M. and Yamagishi, A., A New Method for Known Plaintext Attack of FEAL Cipher, Advances in Cryptology-Eurocrypt'92, Springer-Verlag, pp. 81—91, 1992.

[42] Ohta K. and Aoki, K., Linear Cryptanalysis of the fast Data Encipherment Algorithm, Advances in Cryptology-Crypto'94, Springer-Verlag, pp. 12—16, 1994.

[43] Cusick T. W. and Wood, M. C., The Redoc-II Cryptosystem, Advances in Cryptology-Crypto'90, Springer-Verlag, pp. 545—563, 1991.

[44] Schneier, B., Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons, New York, 1993.

[45] Brown, L., Pieprzyl J. and Seberry, J., Loki: A Cryptographic Primitive for Authentication and Secrecy Applica-

tions, Advances in Cryptology-Auscrypt'90, Springer-Verlag, pp. 229—236, 1990.

[46] Brown, L., Kwan, M., Pieprzyl J. and Seberry, J., Improving Resistance to Differential Cryptanalysis and the Redesign LOKI, Advances in Cryptology-Asiacrypt'91, pp. 36—50, 1993.

[47] Knudsen, L. R., Cryptanalysis of LOKI, Cryptography and Coding II, Oxford, pp. 223—236, Clarendon Press, 1993.

[48] Knudsen, L. R., Cryptanalysis of LOKI 91, Advances in Cryptology-Auscrypt'92, Springer-Verlag, pp. 196—208, 1993.

[49] Tokita, T., Sorimachi T. and Matsui, M., Linear Cryptanalysis of LOKI and S^2 DES, Advances in Cryptology-Asiacrypt'94, Springer-Verlag, pp. 293—303, 1995.

[50] Merkle, R. C., Fast Software Encryption Functions, Advances in Cryptology-Crypto'90, Springer-Verlag, pp. 476—501, 1991.

[51] Gilbert H. and Chauvaud, P., A Chosex Plaintext Attack of the 16-Round Khufu Cryptosystem, Advances in Cryptology-Crypto'94, Springer-Verlag, pp. 359—368, 1994.

[52] Rivest, R. L., The RC5 Encryption Algorithm, In Proceedings of the Workshop on Cryptographic Algorithms, K. U. Leuven, December 1994, Springer-Verlag, pp. 86—96, 1995.

[53] Kaliski B. S., Jr. and Yin, Y. L., On Differential and Linear Cryptanalysis of the RC5 Encryption Algorithm, Advances in Cryptology-Crypto'95, Springer-Verlag, pp. 171—184, 1995.

[54] Lai X. J. and Massey, J. L., A Proposal for a New Block Encryption Standard, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, pp. 389—404, 1991.

[55] Lai, X. J., On the design and Security of Block Ciphers, ETH Series in Information Processing, Vol. 1, Konstanz: Hartung-Gorre Verlag, 1992.

[56] Deamen, J., Govaert R. and Vandewalle, J., Block Ciphers Based on Modular Arithmetic, Proceeding of the 3rd Symposium on State and Progress of Research in Cryptography, Rome, Italy, pp. 80—89, Feb. 1993.

[57] Denning, D. E., The Clipper Encryption System, American Scientist, 81(4): 319—323, July-August 1993.

[58] Massey, J. L., Safer K-64: A Byte-oriented Block-ciphering Algorithm, Fast Software Encryption, December 1993, Springer-Verlag, pp. 1—7, 1994.

[59] Massey, J. L., Safer K-64: On Year Later, Fast Software Encryptiom, December 1994, Springer-Verlag, pp. 212—241, 1995.

[60] Vaudenay, S., On the Need for Multipermutations: Cryptanalysis of MD4 and SAFER, Fast Software Encryption, December 1994, Springer-Verlag, pp. 286—297, 1995.

[61] Knudsen, L. R., A Key-schedule Weakness in SAFERK-64, Advances in Cryptology-Crypto'95, Springer-Verlag, pp. 274—286, 1995.

[62] Davida, G. I. Wells D. L. and Kam, J. B., A Database Encryption System with Subkeys, TR-CS-78-8, Dept. Of Electrical Eng. And Computer Science, The Univ. Of Wisconsin, Milwankee, Wis. (May 1979).

[63] Denning, D. E. R., Cryptography and Data Security, Addison-Wesley, 1982.

[64] Hawkes P. and Connor, L. O., On Applying Linear Cryptanalysis to IDEA, Advances in Cryptology-Asiacrypt'96, Springer Verlag, pp. 105—115, 1996.

[65] Menezes, A. M., P. C. Van Oorschot and S. A. Vanstone, Handbook of Applied Cryptography, New, York, CRC Press, 1996.

[66] Gait, J., A New Nonlinear Pseudorandom Number Generator, IEEE Trans. On SE-3, pp. 359—363, 1977.

[67] Tuchman, W., Hellman Presents no Shortcut Solutions to the DES, IEEE Spectrum Vol. 16, No. 7, pp. 40, 1979.

[68] Konheim, A. G., Cryptography: A Primer, John Wiley & Sons, New York, 1981.

[69] Merkle, R. C. and Hellman, M. E., On the Security of Multiple Encryption, Comm. ACM Vol. 24, No. 7, pp. 465, 1981.

[70] Diffie W. and Hellman, M. E., Exhaustive Cryptanalysis of the NBS Data Encryption Standard, Computer Vol. 10,

No. 6, pp. 74－84, 1977.

[71] Hellman, M. E. ,Commercial Encryption, Network Magazine, Vol. 1, No. 2, pp. 6－10, 1987.

[72] P. C. Van Oorschot and W einer M. J. , A Known-Plaintext Attack on Two-Key Triple Encryption. Advances in Cryptology-Eurocrypt'90, Springer-Verlag, pp. 318－325, 1991.

[73] Campbell K. W. and Wiener, M. J. , Proof That DES Is not a Group, Advances in Cryptology-Crypto'92, Springer-Verlag, pp. 518－526, 1993.

[74] Mauter U. and Massey, J. L. , Cascade Ciphers: The Importance of Being First, 6(1993), pp. 55－61, Journal of Cryptology.

[75] Stinson, D. R. , Cryptography: Theory and practice, CRC Press, 1995.

[76] Simmons, G. J. , Editor. Contemporary, Cryptology, The Science of Information Integrity, IEEE Press, New York, 1992.

[77] Chabaud, F. and Vaudenay, S. , Links Between Differential and Linear Cryptanalysis, Advances in Eurocrypt'94, Springer-Verlag, pp. 356－365, 1995.

[78] Nyberg K. , and Knudsen, L. R. , Provable Security Against Differential Cryptanalysis, Advances in Cryptology-Crypto'92, Springer-Verlag, pp. 566－574, 1993.

[79] Nyberg, K. , Perfect Nonlinear S-box, Advances in Cryptology-Eurocrypt'91, Springer-Verlag, pp. 378－386, 1991.

[80] Nyberg, K. , Differentially Uniform Mappings for Cryptography, Advances in Cryptology-Eurocrypt'93, pp. 55－64, 1994.

[81] Beth T. and Ding, C. , On Almost Perfect Nonlinear Permutations, Advances in Cryptology-Eurocrypt'93, pp. 65－76, 1994.

[82] Feng D. G. and Liu, B. , Almost Perfect Nonlinear Permutations, IEE Electronics, Vol. 30, No. 3: 208－209, 1994.

[83] Biham, E. ,On Matsui's Linear Cryptanalysis, Advances in Cryptology-Eurocrypt'94, pp. 341－355, 1995.

[84] Nyberg, K. ,Linear Approximation of Block Ciphers, Advances in Cryptology-Eurocrypt'94, Spring-Verlag, pp. 439－444, 1995.

[85] Harpes, C. , Kramer G. G. and Massey, J. L. , A Generalization of Linear Cryptanalysis and the Applicability of Matsui's Piling-up Lemma, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, pp. 24－38, 1995.

[86] Zeng, K. C. , Yang J. H. and Dai, Z. D. , Patterns of Entropy Drop of the Key in an S-box of the DES, Advances in Cryptology-Crypto'87, pp. 438－444, 1988.

[87] Yang, J. H. , Dai Z. D. and Zeng, K. C. , Cryptographic Study on S-boxes of DES Type, I, III, System Science and Mathematical Science, Vol. 4, No. 2, 1991, and Vol. 5, No. 1, 1992.

[88] Zeng, K. C. , Yang J. H. and Dai, Z. D. , Cryptographic Study on S-boxes of DES Type, II, System Science and Mathematical Science, Vol. 4, No. 4, 1991.

[89] 王育民、何大可，保密学——基础与应用，西安电子科学技术大学出版社，西安，1990.

[90] 冯登国、宁鹏，S-盒的非线性准则之间的关系，Vol. 19, No. 4, 1998, 72－76 页.

[91] Knudsen L. and Meier, W. , Improved Differential Attacks on RC5, Advances in Cryptology-Crypto'96, Springer-Verlag, pp. 216－228.

[92] Hellman, M. E. , A Cryptanalytic Time-memory Trade-off, IEEE Transactions on Information Theory, 26 (1980), pp. 401－406.

[93] Fiat, A. And Nor, M. , Rigorous Time/Space Trade-Offs for Inverting Functions, In Proceedings of the 23rd Symposium on the Theory of Computing, pp. 534－541, ACM Press, 1991.

第6章　公钥密码算法

在1.1节中我们指出，密码系统根据密钥的特点可分为两类，即私钥密码系统和公钥密码系统。迄今为止，前面讨论的密码系统主要是私钥密码系统，即解密密钥与加密密钥相同或容易从加密密钥导出。在这种系统中，加密密钥的暴露会使系统变得不安全。私钥密码系统的一个严重缺陷是在任何密文传输之前，发送者和接收者必须使用一个安全信道预先通信密钥K。在实际中，达到这一点是很难的。例如，假定发送者和接收者之间的距离很远，他们决定使用电子邮件(e-mail)来通信，在这种情况下，通信双方可能没有一个合理的安全信道。

在公钥密码系统中，解密密钥和加密密钥不同，从一个难于推出另一个，解密和加密是可分离的。通信双方无须事先交换密钥就可建立起保密通信。公钥密码系统的观点是由Diffie和Hellman[1]在1976年首次提出的，它使密码学发生了一场变革。1977年由Rivest，Shamir和Adleman[2]提出了第一个比较完善的公钥密码算法，这就是著名的RSA算法。自从那时起，人们基于不同的计算问题，提出了大量的公钥密码算法。当然，最重要的有RSA算法、Merke-Hellman背包算法、McEliece算法、ElGamal算法和椭圆曲线密码算法等，这些算法将在本章中逐一介绍。

6.1　公钥密码的观点

Diffie和Hellman[1]建议利用计算复杂性设计加密算法。他们指出，NP-完全的问题可作为密码的极好备选方案，因为不可能用已知的技术在多项式时间内求解它们。在计算上比NP中问题更难的问题不适用于加密，因为加密和解密变换必须迅速(即在多项式时间内可计算)。但这意味着密码分析者可以在多项式时间内猜测一个密钥并检验这个解(例如对已知的明文加密)。因此，密码分析者努力破任何多项式时间加密算法必须是NP问题。

Diffie和Hellman[1]曾推测，密码学可以汲取NP复杂性理论，通过检验找出NP-完全问题中可用于密码的问题。信息可通过编码被加密在一个NP-完全问题之中，使得要破译这种密码以普通的方法就要解这个NP-完全问题。但是若使用解密密钥，就可能有捷径求解。为了构造这样的密码，秘密的"陷门"信息必须被嵌在一个涉及单向函数求逆的计算上难的问题之中。一个函数f，如果对它的定义域上的任意x都易于计算$f(x)$，而对于f的值域中的几乎所有的y，即使当f已知时要计算$f^{-1}(y)$也是不可行的，就称f是单向函数(one-way function)。若当给定某些辅助信息下易于计算单向函数f的逆f^{-1}，这就称f是一个陷门单向函数(troopdoor one-way function)。这一辅助信息就是秘密的解密密钥。这就是设计公钥密码体制的基本原理。这类密码的强度取决于它所依据的问题的计算复杂性。然而，计算上困难的问题不一定就意味着是一强密码系统[3,4]，这是因为：(1)复杂性理论通常处理一个问题的单个孤立情况，而密码分析者常常处理大量的统计相关问题的

求解(例如由同一密钥产生的的几条密文);(2)问题的计算复杂性通常是用其最坏情况或平均情况的特性来度量,要是对于密码有用,这一问题必须是在几乎所有情况下是难解的;(3)任意一个困难问题不一定能被变换成一个密码系统,它必须能将陷门信息嵌入到该问题中,使得用这种信息而且只有用这种信息才可能有捷径求解。

值得注意的是,公钥密码体制的安全性是指计算安全性,而绝不是无条件安全性,这是由它的安全性理论基础即复杂性理论决定的。也可用一个例子来说明,假定一个敌手观察到一个密文 y,因为加密变换是公开知道的,所以他用加密变换 E_k 轮流加密每一个可能的明文,直至找到唯一的 x 使得 $y=E_k(x)$ 为止,这个 x 便是 y 的解密。

单向函数在密码学中起一个中心作用。它对公钥密码体制的构造和各种别的内容的研究是很重要的。不幸的是,虽然有许多函数被认为或被相信是单向的,但目前还没有一个函数能被证明是单向的。下面我们给一个被相信是单向函数的例子。

假定 n 是两个大素数 p 和 q 的积,b 是一个正整数,定义 $f:Z_n\to Z_n$,$f(x)=x^b \bmod n$。通常认为它是一个单向函数。

当对 b 和 n 作一个适当的选择时,该函数正是下一节将要讨论的 RSA 算法的加密函数。

值得一提的是,在有些文献中,公钥密码算法的含义很广,不仅包括公钥加密,而且包括各种公钥协议例如数字签名、身份识别协议、密钥交换协议等,但本书中所说的公钥密码算法,除了特别声明外,特指公钥加密算法。

6.2 RSA 算法

6.2.1 RSA 算法的描述

RSA 算法的理论基础是一种特殊的可逆模指数运算。它的安全性是基于分解大整数的困难性。现在我们来描述 RSA 算法。

设 n 是两个不同奇素数 p 和 q 之积,即

$$n=pq, P=C=Z_n, \qquad \varphi(n)=(p-1)(q-1)$$

$$\boldsymbol{K}=\{(n,p,q,a,b)\mid n=pq, p,q \text{ 是素数}, ab\equiv 1 \bmod \varphi(n)\}$$

对每一个 $K=(n,p,q,a,b)$,

定义加密变换为

$$E_K(x)=x^b \bmod n, x\in Z_n \tag{6.2.1}$$

解密变换为

$$D_K(y)=y^a \bmod n, y\in Z_n \tag{6.2.2}$$

公开 n 和 b,保密 p,q 和 a。

为了证明式(6.2.1)定义的加密变换 E_K 和式(6.2.2)定义的解密变换 D_K 满足 $D_K(E_K(x))=x$,对一切 $x\in Z_n$,我们先证明两个有关的结论。

定理 6.2.1(Euler) 对任意的 $a\in Z_n^*$,有

$$a^{\varphi(n)}\equiv 1 \bmod n \tag{6.2.3}$$

其中 $Z_n^*=\{x\in Z_n\mid \gcd(x,n)=1\}$,$\varphi(\cdot)$表示 Euler 函数。

证明：记 $Z_n^*=\{a_1,a_2,\cdots,a_{\varphi(n)}\}$，对任意的 $a\in Z_n^*$，由于存在 a 的逆元 $a^{-1}\in Z_n^*$，所以易证以下集合仍然是 Z_n^*：

$$\{aa_1 \bmod n, aa_2 \bmod n,\cdots,aa_{\varphi(n)} \bmod n\}$$

于是有 $\prod_{i=1}^{\varphi(n)}(aa_i)\equiv\prod_{i=1}^{\varphi(n)}a_i \bmod n$，即 $a^{\varphi(n)}\prod_{i=1}^{\varphi(n)}a_i\equiv\prod_{i=1}^{\varphi(n)}a_i \bmod n$，所以 $a^{\varphi(n)}\equiv 1 \bmod n$。

定理 6.2.2 设 p 和 q 是两个不同的素数，$n=pq$，$\varphi(n)=(p-1)(q-1)$，对任意的 $x\in Z_n$ 及任意的非负整数 k，有

$$x^{k\varphi(n)+1}\equiv x \bmod n \tag{6.2.4}$$

证明：如果 $p\nmid x$，则由定理 6.2.1 可知，$x^{p-1}\equiv 1 \bmod p$，而 $\varphi(p)=(p-1)\mid\varphi(n)$，所以

$$x^{k\varphi(n)+1}\equiv x^{k\varphi(n)}\cdot x\equiv x \bmod p \tag{6.2.5}$$

如果 $p\mid x$，则 $x\equiv 0(\bmod p)$，故式(6.2.5)亦成立。同理可证，恒有

$$x^{k\varphi(n)+1}\equiv x \bmod q \tag{6.2.6}$$

由式(6.2.5)和(6.2.6)及 $\gcd(p,q)=1$ 可知，$x^{k\varphi(n)+1}\equiv x \bmod pq$，即式(6.2.4)成立。

现在我们来验证加密变换 E_K 和解密变换 D_K 满足条件：对一切 $x\in Z_n$，有 $D_K(E_K(x))=x$。因为 $ab\equiv 1 \bmod\varphi(n)$，所以可设 $ab=t\varphi(n)+1$，t 是一个整数且 $t\geqslant 1$。对任意的 $x\in Z_n$，我们有 $D_K(E_K(x))\equiv D_K(x^b)\equiv(x^b)^a\equiv x^{t\varphi(n)+1}\equiv x \bmod n$（由定理 6.2.2）。

这里我们给一个小例子（不安全）。

例 6.2.1 假定用户 B 选择两个素数 $p=5$ 和 $q=7$，因而 $n=pq=35$，$\varphi(n)=(5-1)(7-1)=24$。B 取 $a=11$，然后由 Euclidean 算法求出 $b=a^{-1}\bmod\varphi(n)=11$。B 公开 $n=35$ 和 $b=11$，保密 $p=5$，$q=7$ 和 $a=11$，现在假定另一个用户 A 想把明文 $x=2\in Z_{35}$ 发送给 B。A 计算 $y=E_K(x)\equiv x^b \bmod 35=2^{11} \bmod 35=18$，并将密文 $y=18$ 在公开信道上发送给 B。当 B 收到密文 $y=18$ 时，他使用他的秘密指数 $a=11$ 解密，$y^a \bmod n=18^{11} \bmod 35=2=x$。

因为 RSA 算法的安全性是基于加密函数 $E_K(x)=x^b \bmod n$ 的单向性，所以对敌手来说，解密一个密文是计算上不可行的。允许合法的用户（诸如 B）解密的陷门是分解 $n=pq$ 的知识。因为 B 知道这个分解，所以他能计算 $\varphi(n)=(p-1)(q-1)$，从而能使用 Euclidean 算法计算解密指数 a。

6.2.2 RSA 算法的实现

RSA 算法有许多方面需要讨论，包括建立密码系统的细节、加密和解密的有效性以及安全性问题等。为了建立密码系统，用户 B 需完成下列各步骤：

(1) B 产生两个大素数 p 和 q；

(2) B 计算 $n=pq$ 和 $\varphi(n)=(p-1)(q-1)$；

(3) B 选择一个随机数 $b(0<b<\varphi(n))$ 使得 $\gcd(b,\varphi(n))=1$；

(4) B 使用 Euclidean 算法计算 $a=b^{-1}\bmod\varphi(n)$；

(5) B 将 n 和 b 作为他的公钥直接公开。

我们将逐步讨论 B 如何来完成这些步骤。

密码分析者对该密码系统的一个明显的攻击是企图分解 n。如果能这样做，那么很容易就能计算出 $\varphi(n)=(p-1)(q-1)$，然后像 B 一样能从 b 计算出解密指数 a（人们猜测破译 RSA 是多项式地等价于分解 n，但这一点仍未被证明。所谓两个问题是多项式等价的

意指如果对其中一个问题存在多项式时间算法暗含着对另一个问题也存在多项式时间算法)。因此,如果RSA算法是安全的,那么必须$n=pq$是足够的大,使得分解它是计算上不可行的。目前的分解算法能分解的数已达到130位的十进制数(关于因子分解的详细情况参见6.3节)。因此,基于安全性考虑,建议用户选择的素数p和q大约都为100位的十进制数,那么$n=pq$将是200位的十进制数。RSA的一些硬件实现使用一个512比特长的模。然而,一个512-比特模相当于约154位的十进制数(因为一个整数的二元表示的比特数是十进制数的位数的$\log_2 10$倍),所以从长远的角度来看,512-比特模不能提供足够高的安全性。关于如何选择两个大素数p和q的问题我们将在6.3节中详细讨论。

现在我们来看看加密和解密的算术运算。加密和解密运算都是模n指数运算。因为n是很大的,所以我们必须使用一些有效算法来完成Z_n中的计算,而且需要的计算时间将是依赖于n的二元表示的比特数,即n的长度。

假定n的长度为k,即$k=[\log_2 n]+1$。使用标准的算术技术,不难看出,两个k-比特的整数的加法能在时间$O(k)$内完成,乘法能在时间$O(k^2)$内完成。一个长度至多为$2k$的整数的模n运算能在时间$O(k^2)$内完成(这个过程可通过做长除法完成)。假定$x,y\in Z_n$,那么$xy \bmod n$可按下述方法计算:先计算积xy(是一个长度为$2k$的整数),然后对xy进行模n约化。这两步可在时间$O(k^2)$内完成。我们把这个计算称为模乘法。

下面我们来考虑模指数运算,即形式为$x^c \bmod n$的函数的计算。$x^c \bmod n$的计算可使用$c-1$次模乘法完成,然而如果c是很大的话,那么这种做法是很不有效的。众所周知的"平方一乘"方法是计算模指数的一种有效算法。这种方法计算$x^c \bmod n$至多需要$2l$次模乘法,这里的l是c的长度。因为$l\leqslant k$,所以$x^c \bmod n$能在时间$O(k^3)$内完成。因此,RSA的加密和解密均可在多项式时间内完成(作为k的函数,k是明文或密文的长度)。

"平方-乘"算法的理论依据是,如果

$$b=\sum_{i=0}^{l-1} b_i 2^i \tag{6.2.7}$$

其中$b_i\in\{0,1\}$,$b_{l-1}=1$,则

$$\begin{aligned} x^b &= x^{\sum_{i=0}^{l-1} b_i 2^i} = (x^{b_{l-1}2^{l-2}+\cdots+b_1})^2 \cdot x^{b_0} \\ &= \cdots \\ &= ((\cdots((x^{b_{l-1}})^2 \cdot x^{b_{l-2}})^2 \cdots x^{b_2})^2 \cdot x^{b_1})^2 \cdot x^{b_0} \end{aligned}$$

于是我们有如下的计算$x^b \bmod n$的"平方-乘"算法:

(1)$Z=1$;

(2)for $i=l-1$ downto 0 do;

(a)$Z=Z^2 \bmod n$;

(b)if $b_i=1$ then $Z=Z\times x \bmod n$。

"平方-乘"算法中,在第(2)步中的(a)中总是有l次平方乘完成,在第(2)步中的(b)中进行模乘法的次数等于b的二进制表示中的1的个数,在0与l之间,这样的模乘法的总数至少为l次,至多为$2l$次。

为了提高RSA算法的解密速度,解密时可按下列方式进行:设密文$y\equiv x^b \bmod n$,$n=pq$,设$c_1\equiv c \bmod p$,$c_2\equiv c \bmod q$,$d_1\equiv d \bmod (p-1)$,$d_2\equiv d \bmod (q-1)$,$m_1\equiv m \bmod p$,m_2

$\equiv m \bmod q$，则不难由 c_1, c_2, d_1, d_2 求得 $m_1 \equiv c_1^{d_1} \bmod p$，$m_2 \equiv c_2^{d_2} \bmod q$，由中国剩余定理解同余方程组 $\begin{cases} m \equiv m_1 \bmod p \\ m \equiv m_2 \bmod q \end{cases}$，可得明文 m。此种改进一般可将解密速率提高 4～8 倍。

例 6.2.2 设 $n=35, b=11, x=2$，用"平方-乘"算法计算 x^b。

$b = 2^3 + 2 + 1, b_3 = 1, b_2 = 0, b_1 = 1, b_0 = 1$

i	b_i	Z
3	1	$1^2 \times 2 = 2 \bmod 35$
2	0	$2^2 = 4 \bmod 35$
1	1	$4^2 \times 2 = 32 \bmod 35$
0	1	$32^2 \times 2 = 18 \bmod 35$

目前，RSA 的最有效的硬件实现的加密速率达到每秒 600K 比特（使用一个 512-比特的模 n），与 DES 的硬件实现的每秒 1G 比特的加密速率相比，RSA 大约比 DES 慢了 1500 倍。在软件实现中，DES 大约比 RSA 快 100 倍。这些速率会随着技术的发展而改变，但 RSA 的速率将永远不会达到私钥密码算法的速率。这就是为什么大多数实际系统中仅用 RSA 来变换 DES 密钥，而后用 DES 加密消息的原因。关于 RSA 的硬件实现情况详见文献[6]。

至此，我们讨论了 RSA 的加密和解密运算。在建立 RSA 的各步中，素数 p 和 q 的产生（步骤(1)）将在 6.3 节中讨论；步骤(2)直接地可在时间 $O((\log_2 n)^2)$ 内完成；步骤(3)和(4)已隐含在 Euclidean 算法之中，而 Euclidean 算法的运行时间是 $O((\log_2 n)^2)$，所以步骤(3)和(4)都可在时间 $O((\log_2 n)^2)$ 内完成。

6.2.3 RSA 的安全性分析

本小节我们来谈谈除了分解 n 之外的一些可能的攻击 RSA 的方法。首先我们应注意到，如果密码分析者能计算 $\varphi(n)$，那么他一定能分解 n，从而破译该体制。这是因为我们可通过解下列方程组获得 n 的因子 p 和 q：

$$\begin{cases} n = pq \\ \varphi(n) = (p-1)(q-1) \end{cases}$$

这个方程组可转化为方程 $p^2 - (n - \varphi(n) + 1)p + n = 0$。这个方程的两个根便是 p 和 q，即 n 的因子。因此，一个密码分析者如果能获得 $\varphi(n)$ 的值，那么他就能分解 n，从而破译该系统。换句话说，计算 $\varphi(n)$ 并不比分解 n 容易。

事实上，只要知道 $\varphi(n)$，不需要去分解 n，只要用 Euclidean 算法就可从加密指数 b 计算出解密指数 $a = b^{-1} \bmod \varphi(n)$。

6.2.3.1 解密指数

现在我们来说明一件有趣的事情：计算解密指数 a 的任何算法可作为分解 n 的一个概率算法的子程序或预言(oracle)。这个结果告诉我们，如果 a 被泄露，那么 n 也被危及。选择一个新的加密指数已不能保证系统的安全性，必须重新选择模 n。所以，我们说计算 a 并不比分解 n 容易。然而，这并不排除无须计算 a 可破译该系统的可能性。

我们将要描述的算法是一个 Las Vegas 型概率算法。Las Vegas 算法的精确定义如

下：

定义 6.2.1 假定 $0 \leqslant \varepsilon < 1$ 是一个实数。一个 Las Vegas 算法是一个具有下列性质的概率算法：对问题的任何实例 I，算法以概率 ε 不作出一个回答；然而，如果算法作出一个回答，那么回答一定正确。

一个 Las Vegas 算法可以不给出一个回答，但是它给出的任何回答是正确的。如果我们有一个解决一个问题的 Las Vegas 算法，那么我们反复运行该算法，直到找到一个答案为止。连续运行 m 次，算法将不作出回答的概率为 ε^m。在连续 m 次运行中，算法在第 m 次作出回答的概率为：

$$P_m = \underbrace{\varepsilon \times \varepsilon \times \cdots \times \varepsilon}_{m-1} \times (1-\varepsilon) = \varepsilon^{m-1}(1-\varepsilon)$$

这样，为了获得一个答案，算法必须运行的平均次数或期望次数是

$$\sum_{m=1}^{\infty} m \times P_m = 1/(1-\varepsilon)$$

假定 A 是从加密指数 b 计算解密指数 a 的一个假想的算法。我们将描述一个用 A 作为预言的 Las Vegas 算法，该算法至少以 $\frac{1}{2}$ 的概率分解 n。因此，如果该算法运行 m 次，那么 n 将以至少 $1-1/2^m$ 的概率被分解。该算法是基于 1 模 n 的平方根的某些事实，这里 $n=pq$ 是两个不同的奇素数的乘积。我们知道，同余方程 $x^2 \equiv 1 \bmod p$ 关于模 p 有两个解，即 $x = \pm 1 \bmod p$。类似地，同余方程 $x^2 \equiv 1 \bmod q$ 关于 q 有两个解，即 $x = \pm 1 \bmod q$。因此，1 模 n 有四个平方根，这四个平方根可由中国剩余定理找出。其中两个是 $x = \pm 1 \bmod n$，称为 1 模 n 的平凡的平方根。另两个平方根称为 1 模 n 的非平凡的平方根，其中一个是另一个的模 n 负元。

假定 x 是 1 模 n 的一个非平凡的平方根，则 $n \mid x^2-1=(x-1)(x+1)$，但 n 不能同时整除 $x-1$ 和 $x+1$，所以 $\gcd(x+1,n)=p$（或 q）或 $\gcd(x-1,n)=p$（或 q）。当然，一个最大公因子能使用 Euclidean 算法计算出，而无须知道 n 的分解。因此，一个 1 模 n 的非平凡的平方根的知识使得分解 n 可在多项式时间内完成。这个重要的事实是密码学中许多结果的基础。

现在，我们给出一个使用假想的算法 A 作为一个子程序，通过寻找 1 模 n 的一个非平凡的平方根分解 n 的算法。

给定解密指数 a，分解 n 的算法如下：

(1) 随机地选择 w 使得 $1 \leqslant w \leqslant n-1$；

(2) 计算 $x=\gcd(w,n)$；

(3) 如果 $1<x<n$，那么停止（这时 $x=p$ 或 q，分解 n 成功）；

(4) 计算 $a=A(b)$（A 是一个假想算法，特别地，当对应于 b 的解密指数 a 已知时，无须预言）；

(5) 把 ab 写成 $ab-1=2^t r$，r 为奇数；

(6) 计算 $v=w^r \bmod n$；

(7) 如果 $v \equiv 1 \pmod n$，那么停止（分解 n 失败）；

(8) 当 $v \not\equiv 1 \pmod n$，完成下列各步：

(9) $v_0=v$；

(10) $v=v^2 \bmod n$;

(11) 如果 $v_0 \equiv -1(\bmod n)$，那么停止(分解 n 失败)；否则

(12) 计算 $x=\gcd(v_0+1,n)$(这时 $x=p$ 或 q，分解 n 成功)。

下面我们对上述算法做一点解释。如果我们选到的 w 正好是 p 或 q 的倍数，那么我们能立即分解 n。这一点可在算法的第(2)步中检测出。如果 w 和 n 互素，那么我们通过连续的平方计算 $w^r, w^{2r}, w^{4r}, \cdots$，直到对某一 t 有 $w^{2^t r} \equiv 1(\bmod n)$ 为止。因为 $ab-1=2^s r \equiv 0(\bmod \varphi(n))$，所以我们知道，$w^{2^t r} \equiv 1(\bmod n)$。因此，在至多 s 次迭代后，当圈(当圈指步骤(8)—(10))结束。在当圈末，我们找到一个值 v_0 使得 $v_0^2 \equiv 1(\bmod n)$ 但 $v_0 \neq 1(\bmod n)$。如果 $v_0 \equiv -1(\bmod n)$，那么算法失败；否则，v_0 是 1 模 n 的一个非平凡的平方根，我们便可利用 v_0 分解 n(步骤(12))。在这个算法中，有两种情况出现时不能分解 n：一种是 $w^r \equiv 1(\bmod n)$(步骤(7))；另一种是对某一整数 t，$0 \leqslant t \leqslant s-1$ 使 $w^{2^t r} \equiv -1(\bmod n)$(步骤(11))。但这两种情况出现的概率至多是½，详细推导参见文献[7,8]。这表明该算法分解 n 成功的概率至少是½。

上面关于解密指数的讨论警告 RSA 体制的用户：一旦解密指数泄露，必须重新选择模 n。

6.2.3.2 同模攻击

假定用户 B 有一个 RSA 算法，模为 n，加密指数为 b_1。用户 C 也有一个具有同样模 n 的 RSA 算法，加密指数为 b_2，$\gcd(b_1,b_2)=1$。如果用户 A 想加密同一个明文 x 送给 B 和 C，那么 A 先计算 $y_1=x^{b_1} \bmod n$ 和 $y_2=x^{b_2} \bmod n$，然后将 y_1 发送给 B，将 y_2 发送给 C。假定 O 截取到了 y_1 和 y_2，那么 O 就可按下述步骤计算出 x：

(1) 计算 $c_1=b_1^{-1} \bmod b_2$;

(2) 计算 $c_2=(c_1 b_1-1)/b_2$;

(3) 计算 $x=y_1^{c_1}(y_2^{c_2})^{-1} \bmod n$。

这表明，无论密码系统多么“安全”，O 解密 A 发送的明文是可能的。

在有些应用场合，需要一个可信中心选择一个 RSA 模 n，并对网上的每个用户分配不同的加、解密指数对 (b_i,a_i)。由 6.2.3.1 节的讨论可知，任何一个用户都可用他的加、解密指数对 (b_i,a_i) 的知识分解 n，从而能确定网上的别的用户的解密指数。另外，如果加密同一个明文送给两个或更多的网上的用户，则由上述讨论可知，一个窃听者(不是在网上的任何用户)能利用公开可获得的信息以很高的概率恢复明文。同模攻击告诫人们，不要在不同用户之间共享模 n。

6.2.3.3 选择密文攻击

由 RSA 算法的加密变换易知，对一切 $x_1, x_2 \in Z_n$，有 $E_K(x_1x_2) \equiv E_K(x_1)E_K(x_2) (\bmod n)$，这个性质称为 RSA 算法的同态性质，这是 RSA 算法的一个缺点。这个性质表明，如果敌手知道 c_1 和 c_2 的明文 m_1 和 m_2，他就知道 $c_1c_2 \bmod n$ 所对应的明文是 $m_1m_2 \bmod n$。不过，这个性质可通过引进单向函数来破坏。本小节我们来讨论一下由 RSA 算法的同态性质所诱发的对 RSA 算法的选择密文攻击。

假定一个主动的敌手希望用户 A 为他解密一个特定的密文 $c=m^b \bmod n$。假定 A 除了

c 外他可以为敌手解密任意一个密文。这时，敌手通过选择一个随机整数 $x\in Z_n^*$ 来隐蔽 c，并计算 $\bar{c}=cx^b \bmod n$。敌手一旦将 $\bar{c}$ 提交给 A，A 将为他解密，即计算 $\bar{m}=(\bar{c})^a \bmod n$。因为 $m\equiv(\bar{c})^a\equiv c^a(x^b)^a\equiv mx \bmod n$，所以敌手就能通过公式 $m=\bar{m}x^{-1} \bmod n$ 来计算 m。这种攻击提醒人们，在用 RSA 算法加密时，应先对明文消息进行处理，诸如进行杂凑或通过一个单向变换来破坏 RSA 算法的同态性质。

6.2.3.4 循环攻击

设 $c=m^b \bmod n$ 是一个密文。计算 $c^b \bmod n, c^{b^2} \bmod n, c^{b^3} \bmod n, \cdots, c^{b^i} \bmod n, \cdots$，如果 c^{b^i} 是一个有意义的消息，则有可能 $a=b^{i-1}$；如果 $c^{b^i}=c$，则 $m\equiv c^{b^{i-1}} \bmod n$。这种攻击我们称之为对 RSA 加密的循环攻击。

一个一般的循环攻击将是找到使得 $\gcd(c^u-c,n)>1$ 的最小的正整数 u。如果

$$c^{e^u}\equiv c(\bmod p) \text{ 并且 } c^{e^u}\not\equiv c(\bmod q) \tag{6.2.8}$$

那么，$\gcd(c^{e^u}-c,n)=p$。类似地，如果

$$c^{e^u}\not\equiv c \ (\bmod p) \text{ 并且 } c^{e^u}\equiv c(\bmod q) \tag{6.2.9}$$

那么，$\gcd(c^{e^u}-c,n)=q$。无论在上述哪一种情况下，n 可被分解，从而敌手可破译该系统。另一方面，如果

$$c^{e^u}\equiv c(\bmod p) \text{ 并且 } c^{e^u}\equiv c(\bmod q) \tag{6.2.10}$$

那么 $\gcd(c^{e^u}-c,n)=n$ 并且 $c^{e^u}\equiv c(\bmod n)$。在这种情况下，基本的循环攻击成功，明文 m 可由公式 $m\equiv c^{e^{u-1}}(\bmod n)$ 计算出。因为式(6.2.10)发生的概率要比式(6.2.8)或式(6.2.9)发生的概率要小得多，所以一般的循环攻击在循环攻击成功之前已被终止。这表明，一般的循环攻击本质上可视作是分解 n 的一个算法。而分解 n 被假定是不可行的，所以循环攻击对 RSA 加密的安全性并没有构成威胁。

由上面的讨论可知，满足条件 $m^b\equiv m(\bmod n)$ 的点可能会给密码系统带来一些弱点，这种点称为不动点。我们现在来讨论一下 RSA 算法的不动点的个数。

考虑方程 $x^m\equiv 1 \bmod n$ 的解的个数，$n=pq$，p 和 q 为素数。由初等数论可知，$x^m\equiv 1 \bmod p$ 恰有 $d_1=\gcd(m,p-1)$ 个解。类似地，$x^m\equiv 1 \bmod q$ 也恰有 $d_2=\gcd(m,q-1)$ 个解。由中国剩余定理可知，$x^m\equiv 1 \bmod n$ 的解的个数为 d_1d_2。

因为 $x^{b-1}\equiv 1 \bmod p$ 有 $d_1=\gcd(b-1,p-1)$ 个解，所以 $x^b\equiv x \bmod p$ 有 $1+d_1$ 个解。同理，$x^b\equiv x \bmod q$ 有 $1+d_2=1+\gcd(b-1,q-1)$ 个解。故由中国剩余定理可知，$x^b\equiv x \bmod n$ 有 $(1+d_1)(1+d_2)=(1+\gcd(b-1,p-1))(1+\gcd(b-1,q-1))$ 个解，这也正是 RSA 算法的不动点的个数。因为 $b-1$，$p-1$，$q-1$ 都是偶数，所以 RSA 算法至少有 9 个不动点。一般认为，在实际中，RSA 算法的不动点相对而言很少，对其安全性没有构成威胁。

关于 RSA 算法的安全性需要考虑许多方面，除了上述所提到的之外，我们还必须注意两个素数 p 和 q 不能选择的太接近，否则密码分析者可先计算 $\sqrt{n}$，然后在 $\sqrt{n}$ 的附近搜索 p 和 q 来分解 n。关于 p 和 q 的选取我们可以通过 3.3 节所介绍的因子分解方法来吸取经验。另外，加密指数 b 和解密指数 a 都不能选择的太小，虽然这可带来加密速度快、容易计算等优点，但不幸的是，文献[9]和[10]中给出了这种情况的成功攻击。下面一小节我们来讨论 RSA 算法的安全性的另一方面，即 RSA 加密对明文的信息泄露问题。

6.2.4 关于明文比特的部分信息

本小节我们主要来关心RSA算法中密文所泄露的有关明文比特的部分信息。在这里我们只考虑下列两种部分信息：

(1) 给定 $y=E_K(x)$，计算 Parity(y)，这里 Parity(y)表示 x 的最低位比特；

(2) 给定 $y=E_K(x)$，计算 half(y)，这里

$$\text{half}(y)=\begin{cases}0 & 0\leqslant x<n/2\\ 1 & n/2<x\leqslant n-1\end{cases}$$

下面我们将证明，给定 $y=E_K(x)$，计算 Parity(y)或 half(y)的任何算法可用作构造计算明文 x 的算法的一个预言。这意味着，给定一个密文，计算明文的低位比特是多项式等价于确定整个明文。

首先，我们说明计算 Parity(y)是多项式等价于计算 half(y)。这一事实，可由下述三个等式获得：

$$E_K(x_1x_2)=E_K(x_1)E_K(x_2) \tag{6.2.11}$$

$$\text{half}(y)=\text{Parity}(y\times E_K(2)\bmod n) \tag{6.2.12}$$

$$\text{Parity}(y)=\text{half}(y\times E_K(2^{-1})\bmod n) \tag{6.2.13}$$

由RSA算法中的加密函数 $E_K=x^b\bmod n$ 易知，式(6.2.11)成立。式(6.2.12)和式(6.2.13)的证明是类似的，我们只证明式(6.2.13)。如果 Parity$(y)=0$，说明 y 对应的明文 $x(0\leqslant x\leqslant n-1)$是偶数，从而 $\frac{x}{2}$ 是整数，且 $0\leqslant\frac{x}{2}\leqslant\frac{n-1}{2}$。而 $y\times E_K(2^{-1})\bmod n=E_K\left(\frac{x}{2}\right)$，故 $\text{half}(y\times E_K(2^{-1})\bmod n)=\text{half}\left(E_K\left(\frac{x}{2}\right)\right)=0$。如果 Parity$(y)=1$，说明 y 对应的明文 $x(0\leqslant x\leqslant n-1)$是奇数，可设 $x=2t+1$，$0\leqslant t<\frac{n-2}{2}$。而 $2^{-1}\bmod n=\frac{n+1}{2}$，所以

$$y\times E_K(2^{-1})\bmod n=E_K(x\cdot 2^{-1})=E_K\left((2t+1)\times\frac{n+1}{2}\right)$$

$$=E_K\left(\frac{2tn+2t+n+1}{2}\right)=E_K\left(t+\frac{n+1}{2}\right)$$

显然

$$\frac{n+1}{2}\leqslant t+\frac{n+1}{2}<\frac{n-2}{2}+\frac{n+1}{2}=n-\frac{1}{2}$$

故 $\text{half}(y\times E_K(2^{-1})\bmod n)=1$。综上所述，式(6.2.13)成立。

现在的问题是，给定一个计算 half(y)的假想的算法，如何计算 $x=D_K(y)$。下面我们来描述解决这一问题的一个算法，其具体运行过程如下：

(1) 表示 $k=[\log_2 n]$（$[x]$表示不大于 x 的最大整数，比如$[3.5]=3$）；

(2) 对 $i=0$ 到 k，计算

(3) $y_i=\text{half}(y)$；

(4) $y=(y\times e_K(2))\bmod n$；

(5) $l_0=0$；

(6) $h_i=n$；

(7) 对 $i=0$ 到 k，计算

(8) mid$=(h_i+l_0)/2$;

(9) 如果 $y_i=1$,那么 $l_0=$mid;否则,$h_i=$mid;

(10)$x=[h_i]$

我们对这个算法稍作解释。在步骤(2)~(4),我们计算出 $y_i=\text{half}(y\times(E_K(2)^i)=\text{half}(E_K(x\times 2^i))$,$0\leqslant i\leqslant\log_2 n$。我们可以看到

$$\text{half}(E_K(x))=0\Leftrightarrow x\in\left[0,\frac{n}{2}\right)$$

$$\text{half}(E_K(2x))=0\Leftrightarrow x\in\left[0,\frac{n}{4}\right)\cup\left[\frac{n}{2},\frac{3n}{4}\right)$$

$$\text{half}(E_K(4x))=0\Leftrightarrow x\in\left[0,\frac{n}{8}\right)\cup\left[\frac{n}{4},\frac{3n}{8}\right)\cup\left[\frac{n}{2},\frac{5n}{8}\right)\cup\left[\frac{3n}{4},\frac{7n}{8}\right)$$

$$\cdots$$

因此,我们能通过二元搜索技术(步骤(7)~(10))找到 x。

例 6.2.3 设 $n=35$,$b=11$,我们有一个密文 $y=12$。易知,$E_K(2)=18$。假设对 half 使用我们的预言,我们可在算法的第(3)步获得下列的值 y_i:

i	0	1	2	3	4	5
y_i	0	0	0	1	0	1

那么搜索明文 x 的二元搜索过程如下:

i	l_0	mid	h_i
0	0.00	17.50	35.00
1	0.00	8.75	17.50
2	0.00	4.38	8.75
3	0.00	2.19	4.38
4	2.19	3.29	4.38
5	2.19	2.74	3.28
	2.74	3.02	3.29

因此,明文是 $x=[3.02]=3$。

关于 RSA 算法的比特安全性的更详细的讨论,感兴趣的读者可参阅文献[11,12]。另外,文献[66]给出了 RSA 的一种变形,称作 Batch RSA。

RSA 算法的专利:

RSA 算法只有在美国申请了专利,美国专利将于 2000 年 9 月 20 日到期。

6.3 素性检测和因子分解

6.3.1 素性检测

我们知道,在 RSA 算法的建立中产生大的"随机素数"是必不可少的步骤。那么如何

来产生大的“随机素数”呢？在实际中，往往是首先产生大的随机数，然后使用一个概率多项式时间算法测试它的素性。本小节我们主要来介绍两个测试随机数的素性的概率算法，即 Solovay-strassen 算法和 Miller-Rabin 算法。这两个算法测试整数 n 的素性都可在 $\log_2 n$ 的多项式时间内完成，但是这些算法有可能把一个合数判定为素数。然而，运行算法足够多次后，错误概率可降低到任何事先所期望的值以下。

现在我们担心两件事情：一件是我们会不会把素数用完；另一件是会不会有两个人偶然地选择了同样的素数呢？数论中的著名的素数定理为我们解除了这些后顾之忧，它告诉我们不超过 N 的素数大约有 $N/\ln N$ 个，而且素数有无限多个。因此，如果两个人都随机地选择一个不超过 N 的素数，那么他们选择同样的素数的概率大约为 $(\ln N/N)^2$。当 N 充分大时，$(\ln N/N)^2$ 趋于 0，所以没必要担心两个人选择同样的素数这种情况发生，也没必要担心素数会被用完。

由素数定理知，如果 p 是随机选择的一个数，那么 p 是素数的概率大约为 $(p/\ln p)/p=1/\ln p$。对一个 512 比特的模来说，我们有 $1/\ln p\approx 1/177$。也就是说，平均来说，适当长度的 177 个随机整数 p 中有一个是素数（当然，如果限制 p 是奇整数，那么概率大约为 $2\times 1/177=2/177$）。由此可见，产生充分大的“概率素数”是切实可行的，因此建立 RSA 算法也是切合实际的。

下面我们将要介绍的概率算法实际上都是 Monte Carlo 型算法，现在我们来给出这个概念的精确定义。

定义 6.3.1 对一个确定问题的一个偏是的（yes-biased）Monte Carlo 算法是具有下列性质的一个概率算法：一个“是”回答总是正确的，但一个“否”回答也许是不正确的。类似地，可定义一个偏否的（no-biased）Monte Carlo 算法。我们说一个偏是的 Monte Carlo 算法具有错误概率 ε，如果算法对任何回答应该是“是”的实例将至多以 ε 给一个不正确的回答“否”。

一个 Monte Carle 算法总是给一个回答，但回答也许是不正确的。而一个 Las Vegas 算法也许不给一个回答，但任何回答总是正确的。

我们首先来描述 Solovay-Strassen 算法，这个算法是对合数问题的一个偏是的 Monte Carlo 算法，该算法具有 1/2 的错误概率。因此，如果算法回答“是”，那么 n 是合数；反之，如果 n 是合数，那么该算法至少以 1/2 的概率回答“是”。

对一个奇整数 n 的 Solovay-Strassen 素性测试算法：

(1) 选择一个随机整数 a，$1\leqslant a\leqslant n-1$；

(2) 如果 $\left(\frac{a}{n}\right)\equiv a^{(n-1)/2}(\mathrm{mod}\, n)$，那么回答“$n$ 是素数”；否则，回答“n 是合数”。

我们已经知道，计算 $a^{(n-1)/2}\mathrm{mod}\, n$ 可在时间 $O((\log_2 n)^3)$ 内完成，而 Jacobi 符号 $\left(\frac{a}{n}\right)$ 可在时间 $O((\log_2 n)^2)$ 内完成，所以 Solovay-Strassen 素性测试算法是一个多项式时间概率算法。关于用到的一些数论知识参见本书附录或任何一本数论教科书。显然，这个算法是对合数问题的一个偏是的 Monte Carlo 算法。至于该算法具有错误概率 1/2 的详细证明，参见文献[8,13,14]。

假定我们已经产生了一个随机数 n 并且用 Solovay-Strassen 算法测试它的素性。如果我们运行该算法 m 次，n 是素数的概率将随着 m 的增大而迅速增大，增加的具体数量关

系参见文献[8]。

本小节最后我们来描述另一个对合数问题的 Monte Carlo 算法，即 Miller-Rabin 算法。

对一个奇整数 n 的 Miller-Rabin 素性测试算法：

(1) 令 $n-1=2^k m$，m 是奇数；

(2) 选择一个随机整数 a，$1\leqslant a\leqslant n-1$；

(3) 计算 $b\equiv a^m \bmod n$；

(4) 如果 $b\equiv 1 \pmod n$，那么回答"n 是素数"并停止；

(5) 对 $i=0$ 到 $k-1$，做

(6) 如果 $b\equiv -1 \pmod n$，那么回答"n 是素数"并停止；否则，计算 $b\equiv b^2 \bmod n$；

(7) 回答"n 是合数"。

显然，Miller-Rabin 算法是一个多项式时间算法，它的时间复杂度为 $O((\log_2 n)^3)$。事实上，Miller-Rabin 算法比 Solovay-Strassen 算法快。该算法是对合数问题的一个偏是的 Monte Carlo 算法，其错误概率至多为 1/4，详细证明参见文献[8,15,16]。

产生大素数有多种方法，上述我们只介绍了两种方法。关于这个领域的一些进展可参阅文献[17,18]。想多了解一些产生大素数的算法的读者可查阅文献[19]。

6.3.2 因子分解

RSA 算法的安全性是基于分解大整数的困难性。如果密码分析者能从用户的公钥 n ($=pq$)在多项式时间内求出 p 和 q，那么 RSA 算法将是不安全的。正因为如此，近年来人们对因子分解这一古老的数论问题颇感兴趣并取得了一些可喜的进展[20,21,22]。讨论因子分解算法的文献很多，我们不可能在这里逐一介绍。但我们尽力给一个简洁的综述，主要包括目前最好的因子分解算法的一个非正式的讨论和它们在实际中的使用。目前，最有实用价值的因子分解算法有二次筛法[23,24,25]、数域筛法[21,26]、椭圆曲线算法[28,29]、Pollard 的 rho-方法和 p-1 算法[30]、连分式算法[31,32]、试除法、Williams 的 $p+1$ 算法等。其中最有效的三种算法是二次筛法(quadratic sieve)，椭圆曲线算法(elliptic curve algorithm)和数域筛法(number field sieve)。

本小节我们假定所要分解的 n 是奇数。对比较小的 n 来说，比如 $n<10^{12}$，通过试除 1 与$[\sqrt{n}]$之间的每一个奇整数完全可分解 n，这种方法就是试除法(trial division)。但对比较大的 n 来说，这种方法已不可行，一般地，我们需要使用更复杂的技术。

值得注意的是，计算模 n 的平方根和因子分解这两个问题在计算上是等价的。这一事实可由 6.2.3.1 节中的讨论容易得出。

6.3.2.1 $p-1$ 方法

现在我们来描述 Pollard 在 1974 年提出的分解整数的 $p-1$ 算法。

$p-1$ 分解算法：

输入：n(奇数)和 B。

(1) $a=2$；

(2) 对 $j=2$ 到 B，计算 $a\equiv a^j \bmod n$；

(3) $d=\gcd(a-1,n)$;

(4) 如果 $1<d<n$,那么 d 是 n 的一个因子(分解成功);否则,没有找到 n 的因子(分解失败)。

下面我们来对上述算法的合理性稍作解释。

假定 p 是 n 的一个素因子。若对每一个素数幂 $q|(p-1)$,有 $q\leqslant B$,则必有 $(p-1)|B!$。在第(2)步末,$a\equiv 2^{B!}(\mathrm{mod}n)$,因为 $p|n$,所以 $a\equiv 2^{B!}(\mathrm{mod}p)$。由定理 6.2.1 可知,$2^{p-1}\equiv 1(\mathrm{mod}p)$。因为 $(p-1)|B!$,所以 $a\equiv 1(\mathrm{mod}p)$(第(3)步)。这样在第(4)步有 $p|(a-1)$且 $p|n$,所以 $p|d=\gcd(a-1,n)$。在第(3)步除了 $a=1$ 以外,整数 d 总是 n 的一个非平凡因子。一旦找到 n 的一个非平凡因子 d,然后我们将对 d 和 n/d 继续分解(如果 d 和 n/d 还是合数)。

在 $p-1$ 算法中,有 $B-1$ 个模指数,利用"平方-乘"算法计算每一个模指数需要至多 $2\log_2 B$ 个模乘法。gcd 的计算可用 Euclidean 算法在时间 $O((\log_2 n)^3)$ 内完成。因此,该算法的时间复杂度是 $O(B\log_2 B(\log_2 n)^2+(\log_2 n)^3)$。如果 B 是 $O((\log_2 n)^i)$,i 是某一固定整数,那么该算法的确是多项式时间的。然而,对 B 的这样的一个选择,算法成功的概率将是很小的。另一方面,如果猛增加 B 的尺寸,比如说 $\sqrt{n}$,那么该算法成功的概率将是很高的,但是它并不比试除法快。可见,这个算法的缺陷是它要求 n 有一个素因子 p 使得 $p-1$ 只有小的素因子。这就要求我们在构造 RSA 模 $n=pq$ 时选择的 p 和 q 应使 $p-1$ 和 $q-1$ 含有大的素因子,使 $p-1$ 分解算法失败。一般的方法是选择两个大素数 p_1 和 q_1,使得 $p=2p_1+1$ 和 $q=2q_1+1$ 也是素数(形为 $p=2p_1+1$ 的素数通常称为安全素数,其中 p_1 为素数)。此时,RSA 模 $n=pq$ 将能抵抗 $p-1$ 方法的分解。

80 年代中期由 Lenstra 提出的更有效的椭圆曲线算法实际上是 $p-1$ 方法的一般化。我们在这里不讨论这种方法的理论,但我们提醒人们值得注意的是椭圆曲线算法的成功率依赖于接近于 p 的一个整数只有小的素因子。

6.3.2.2 Dixon 算法和二次筛法

Dixon 算法的理论依据是下列的简单事实:如果 $x\neq\pm y(\mathrm{mod}n)$ 且 $x^2\equiv y^2(\mathrm{mod}n)$,则 $\gcd(x-y,n)$ 是 n 的一个非平凡因子。该算法的基本观点是,首先建立一个所谓的因子基(factor base),因子基是小素数的一个集合 $\boldsymbol{B}$。其次,找到一些整数 x 使得 $x^2\mathrm{mod}n$ 的所有素因子都在因子基 $\boldsymbol{B}$ 之中。最后,将某些 x 相乘使得每一个在因子基中的素数出现偶数次。这样我们就建立起了一个所期望的类型的同余方程 $x^2\equiv y^2(\mathrm{mod}n)$,该方程可能导致 n 的一个分解。

现在我们用一个例子来说明 Dixon 算法的基本观点。

例 6.3.1 假定 $n=15770708441$,$\boldsymbol{B}=\{2,3,5,7,11,13\}$。考虑三个同余方程:

$$8340934156^2\equiv 3\times 7(\mathrm{mod}n)$$

$$12044942944^2\equiv 2\times 7\times 13(\mathrm{mod}n)$$

$$2773700011^2\equiv 2\times 3\times 13(\mathrm{mod}n)$$

如果将上述三个同余方程相乘,我们将得到

$$(8340934156\times 12044942944\times 2773700011)^2\equiv(2\times 3\times 7\times 13)^2(\mathrm{mod}n)$$

即 $9503435785^2\equiv 546^2(\mathrm{mod}n)$。则

$$\gcd(9503435785-546,15770708441)=115759$$

这样，我们就找到了 n 的一个因子 115759。

假定 $\boldsymbol{B}=\{p_1,p_2,\cdots,p_B\}$ 是因子基。C 比 B 稍大一点，比如说，$C=B+10$，假定我们已经获得了 C 个同余式

$$x_j^2\equiv p_1^{\alpha_{1j}}\times p_2^{\alpha_{2j}}\times\cdots\times p_B^{\alpha_{Bj}}(\mathrm{mod}n)\qquad 1\leqslant j\leqslant C$$

对每一个 j，考虑向量 $a_j=(\alpha_{1j}\mathrm{mod}2,\cdots,\alpha_{Bj}\mathrm{mod}2)\in Z_2^B$。如果我们能找到 $\{a_j\}_{j=1}^C$ 的一个子集使得其模 2 和是向量 $(0,0,\cdots,0)$，那么相应的 x_j 的乘积的每一个在 B 中的因子出现偶数次。

例 6.3.1(续)　设

$$a_1=(0,1,0,1,0,0)$$
$$a_2=(1,0,0,1,0,1)$$
$$a_3=(1,1,0,0,0,1)$$

易知：$a_1+a_2+a_3=(0,0,0,0,0,0)\mathrm{mod}2$。这里 $C=3<B=6$。

可见，找一个 C 个向量 $a_1,a_2,\cdots,a_C$ 的子集使其模 2 和为零向量不外乎是找一组在 Z_2 上线性相关的向量。如果 $C>B$，这样的一个线性相关组一定存在，并且可用高斯消去法容易找到。只所以选取 $C>B+1$，是因为并非任何给定的同余式 $x^2\equiv y^2(\mathrm{mod}n)$ 都能导致 n 的分解。大约有 1/2 的概率使得 $x\equiv\pm y(\mathrm{mod}n)$。但是，如果 $C>B+1$，那么我们可以从 $\{a_j\}_{j=1}^C$ 中选取许多不同的线性相关组，从而获得许多同余式 $x^2\equiv y^2(\mathrm{mod}n)$，这就大大地增加了分解 n 的机会。

尚须讨论的是我们如何来获得这些整数 x_j 使得 $x_j^2\mathrm{mod}n$ 在因子基 $\boldsymbol{B}$ 上完全地分解 n。现有许多方法处理这个问题，但一个通用的方法是 Pomerance 提出的二次筛法，该方法使用了形式为 $x_j=j+[\sqrt{n}\,](j=1,2\cdots)$ 的整数。详细讨论请参阅文献[23]。

$B=|\boldsymbol{B}|$ 的最优选择近似为 $\sqrt{e^{\ln n\ln\ln n}}$，在 B 的这种选择下，分解 n 的平均运行时间为 $O\left(e^{(1+o(1))\sqrt{\ln n\ln\ln n}}\right)$。其中符号 $o(1)$ 表示当 $n\to\infty$ 时，n 的一个函数趋近于 0。

数域筛法是 80 年代后期发展起来的一个分解算法。它通过构造一个同余式 $x^2\equiv y^2(\mathrm{mod}n)$ 和代数整数环中的计算来分解 n。二次筛法，椭圆曲线算法和数域筛法的平均运算时间分别为：

二次筛法　$O\left(e^{(1+o(1))\sqrt{\ln n\ln\ln n}}\right)$

椭圆曲线算法　$O\left(e^{(1+o(1))\sqrt{2\ln p\ln\ln p}}\right)$

数域筛法　$O\left(e^{(1.92+o(1))(\ln n)^{1/3}(\ln\ln n)^{2/3}}\right)$

其中 p 表示 n 的最小素因子。

在最坏的情况下，$p\approx\sqrt{n}$，二次筛法和椭圆曲线算法的平均运行时间本质上一样。但在这种情况下，二次筛法一般优于椭圆曲线算法。如果 n 的素因子具有不同的长度，那么椭圆曲线算法是更有效的。Brent 在 1988 年使用椭圆曲线方法分解了一个很大的 Fermat 数 $2^{2^{11}}+1$。

二次筛法是目前分解 RSA 模(所谓 RSA 模 n 是指 $n=pq$，p 和 q 是两个不同的素数，p 和 q 的长度大致相等)的最有效的算法。1983 年，Davis 等人利用二次筛法成功地分解了一个 69 位的十进制整数，该数是 $2^{251}-1$ 的一个合数因子。80 年代起人们不断地使用二

次筛法进行整数分解。1989 年，Lenstra 和 Manasse 利用二次筛法并且通过把计算分配给成百个工作站的办法成功地分解了一个 106 位的十进制整数。最近，1994 年 4 月，Atking 等人使用二次筛法分解了一个称作 RSA-129 的 129 位的十进制整数。

数域筛法是这三个算法中最新的算法。似乎该算法有很大的潜力，因为它的平均运行时间比二次筛法和椭圆曲线算法的平均运行时间都少。数域筛法仍然处在发展阶段，但人们推测该算法对大于 125～130 位的十进制数也许可证明是较快的算法。1990 年，Lenstra 等人[27]利用数域筛法把 $2^{2^9}+1$ 分解成三个素数，这三个素数分别是 7 位、49 位和 99 位十进制数。

6.4 ElGamal 算法和离散对数

ElGamal 算法[33]是在密码协议中有着大量应用的一类公钥密码算法，它的安全性是基于离散对数问题(discrete logarithm problem)的困难性。离散对数问题也就是子群成员问题。在一个有限域 Z_p(p 为素数)上的离散对数问题可叙述为：给定一个素数 p 和 Z_p 的一个本原元 α，对 $\beta\in Z_p^*$，找唯一的一个整数 a，$0\leqslant a\leqslant p-2$，使得 $\alpha^a\equiv\beta(\mathrm{mod}\,p)$。通常用 $\log_\alpha\beta$ 来表示 a。一般地，如果仔细选择 p，那么认为该问题是困难的。特别是，目前还没有找到计算离散对数问题的多项式时间算法。为了抵抗已知的攻击，p 应该至少是 150 位十进制整数，并且 $p-1$ 至少有一个大的素因子。之所以离散对数问题在密码环境中是有用的，是因为找离散对数是困难的，但模指数逆运算可使用“平方-乘”方法有效地计算。换句话说，对适当的素数 p，模 p 指数运算是一个单向函数。本节我们主要来介绍 ElGamal 密码体制和关于离散对数问题的一些算法。

6.4.1 ElGamal 算法

ElGamal 密码算法是非确定性的，因为密文依赖于明文 x 和加密者选择的随机值 k。所以，同样的明文可被加密成许多密文。现在我们来详细地描述 Z_p^* 上的 ElGamal 公钥密码算法。

设 p 是一个素数并且离散对数问题在 Z_p 上是难处理的，$\alpha\in Z_p^*$ 是一个本原元。$\boldsymbol{P}=Z_p^*$，$\boldsymbol{C}=Z_p^*\times Z_p^*$，$\boldsymbol{K}=\{(p,\alpha,a,\beta)\mid\beta\equiv\alpha^a(\mathrm{mod}\,p)\}$。值 p，α 和 β 是公开的，a 是保密的。

对 $K=(p,\alpha,a,\beta)$和一个秘密地随机选择的数 $k\in Z_{p-1}$，定义 $E_K(x,k)=(y_1,y_2)$，这里 $y_1=\alpha^k\mathrm{mod}\,p$，$y_2=x\beta^k\mathrm{mod}\,p$。对 $y_1,y_2\in Z_p^*$，定义

$$D_K(y_1,y_2)=y_2(y_1^a)^{-1}\mathrm{mod}\,p$$

易知对每一个固定的 $K\in\boldsymbol{K}$，对一切 $x,k\in Z_p^*$，有 $D_K(E_K(x,k))\equiv x \bmod p$。

下面我们用一个例子来说明一下 ElGamal 密码算法的加、解密过程。

例 6.4.1 假定 $p=2579$，$\alpha=2$，$a=765$，因此 $\beta=2^{765}\bmod 2579=949$。现在假定 A 想发送消息 $x=1299$ 给 B。比如说 $k=853$ 是 A 随机选择的整数。A 计算 $y_1=2^{853}\bmod 2579=435$，并用 β^k 来隐蔽 x 产生 y_2，即 $y_2=1299\times 949^{853}\bmod 2579=2396$。A 将密文 $y=(435,2396)$传送给 B，B 收到密文 y 后，计算 $x=2396\times(435^{765})^{-1}\bmod 2579=1299$。

6.4.2 求离散对数问题的算法

在这一小节中，我们假定 p 是素数并且 α 是 Z_p 的一个本原元。我们把 p 和 α 取定，这

时离散对数问题可陈述为：给定 $\beta\in Z_p^*$，找唯一的一个指数 a，$0\leqslant a\leqslant p-2$，使得 $\alpha^a\equiv\beta(\mathrm{mod}\ p)$。

显然，离散对数问题可在时间 $O(p)$ 内和空间 $O(1)$ 内用穷搜索方法解决。通过预计算所有可能的值 α^a，并按有序对 (a,α^a) 的第二个坐标进行排序，我们能在 $O(1)$ 时间内解决离散对数问题，但所需的预计算时间为 $O(p)$，存贮空间为 $O(p)$。这些都是平凡的算法。计算离散对数问题的非平凡算法有很多[34,35]，这里只介绍三种，即 Shanks 算法[8]、Pohlig-Hellman 算法[36]和指数计算(index calculus)方法[8]。

6.4.2.1 Shanks 算法

Shanks 算法是一种时间-空间折衷法。令 $m=\left[\sqrt{p-1}\right]$，计算离散对数问题的 Shanks 算法如下：

(1) 计算 $\alpha^{mj}\mathrm{mod}\ p$，$0\leqslant j\leqslant m-1$；

(2) 把 m 个有序对 $(j,\alpha^{mj}\mathrm{mod}\ p)$ 按第二个坐标排序，获得一张表 L_1；

(3) 计算 $\beta\alpha^{-i}\ \mathrm{mod}\ p$，$0\leqslant i\leqslant m-1$；

(4) 把 m 个有序对 $(i,\beta\alpha^{-i}\mathrm{mod}\ p)$ 按第二个坐标排序，获得一张表 L_2；

(5) 找一对 $(j,y)\in L_1$ 和一对 $(i,y)\in L_2$，即找第二个坐标相同的两个对；

(6) 定义 $\log_\alpha\beta=mj+i\mathrm{mod}(p-1)$。

现在我们对 Shanks 算法作一点解释。如果需要的话，算法中的第(1)步和第(2)步可预计算，这些计算不影响算法的平均运行时间。如果 $(j,y)\in L_1$，$(i,y)\in L_2$，那么 $\alpha^{mj}=y=\beta\alpha^{-i}$，所以 $\alpha^{mj+i}=\beta$，这正是我们所期望的。反之，对任何 β，我们可令 $\log_\alpha\beta=mj+i$，这里 $0\leqslant j,i\leqslant m-1$。因此，算法中的步骤(5)的搜索总是成功的。

易知，Shanks 算法的运行时间和存贮空间均为 $O(m)$。

下面我们用一个例子来说明一下 Shanks 算法的具体运行过程。

例 6.4.2 假定 $p=809$，$\alpha=3$，$\beta=525$，我们希望找到 $\log_\alpha\beta$。令 $m=\left[\sqrt{808}\right]=29$，那么 $\alpha^{29}\mathrm{mod}809=99$。

首先计算有序对 $(j,99^j\mathrm{mod}809)$，$0\leqslant j\leqslant 28$，我们获得下表：

(0,1)	(1,99)	(2,93)	(3,308)	(4,559)
(5,329)	(6,211)	(7,664)	(8,207)	(9,268)
(10,644)	(11,654)	(12,26)	(13,147)	(14,800)
(15,727)	(16,781)	(17,2164)	(18,632)	(19,275)
(20,528)	(21,496)	(22,564)	(23,15)	(24,676)
(25,586)	(26,575)	(27,295)	(28,81)	

整理上表可得 L_1。

其次计算有序对 $(i,525\times(3^i)^{-1}\ \mathrm{mod}\ 809)$，$0\leqslant j\leqslant 28$，我们获得下表：

(0,525)	(1,175)	(2,328)	(3,379)	(4,396)
(5,132)	(6,44)	(7,554)	(8,724)	(9,511)
(10,440)	(11,686)	(12,768)	(13,256)	(14,355)
(15,388)	(16,399)	(17,133)	(18,314)	(19,644)
(20,754)	(21,521)	(22,713)	(23,777)	(24,259)
(25,356)	(26,658)	(27,489)	(28,163)	

整理上表可得 L_2。

现在，我们同时查表 L_1 和 L_2，可找到$(10,644)\in L_1$，$(19,644)\in L_2$。因此，我们能计算 $\log_3 525=29\times 10+19=309$。我们来验证一下，的确有 $3^{309}\equiv 525 \pmod{809}$。

6.4.2.2 Pohlig-Hellman 算法

假定 $p-1=\prod_{i=1}^{K} p_i^{C_i}$，这里 p_i 是不同的素数。值 $a=\log_\alpha \beta$ 模 $p-1$ 唯一确定。如果我们对每一个 $i(1\leqslant i\leqslant K)$能计算 $a \bmod p_i^{C_i}$，那么我们可利用中国剩余定理来计算 $a \bmod (p-1)$。因此，让我们假定 q 是满足下列条件的一个素数：$p-1\equiv 0 \pmod{q^C}$但 $p-1\not\equiv 0 \pmod{q^{C+1}}$。下面我们来说明如何计算值 $x=a \bmod q^C$，$0\leqslant x\leqslant q^C-1$。先将 x 表示成 q 进制数，$x=\sum_{i=0}^{C-1} a_i q^i$，$0\leqslant a_i\leqslant q-1$，$0\leqslant i\leqslant C-1$。再将 a 表示成 $a=x+q^C S$，S 是某一整数。

算法的第一步是计算 a_0。因为 $\beta^{(p-1)/q}\equiv \alpha^{(p-1)(x+q^C S)/q} \pmod p$，所以 $\beta^{(p-1)/q}\equiv \alpha^{(p-1)a_0/q} \pmod p$，当且仅当 $\alpha^{(p-1)(x+q^C S)/q}\equiv \alpha^{(p-1)a_0/q} \pmod p$，当且仅当

$$(p-1)(x+q^C S)/q \equiv (p-1)a_0/q \pmod{(p-1)}$$

而

$$\frac{(p-1)(x+q^C S)}{q}-\frac{(p-1)a_0}{q}=\frac{p-1}{q}(x+q^C S-a_0)=\frac{p-1}{q}\left(\sum_{i=1}^{C-1} a_i q^i+q^C S\right)$$

$$=(p-1)\left(\sum_{i=1}^{C-1} a_i q^{i-1}+q^{C-1}S\right)\equiv 0 \pmod{(p-1)}$$

故我们证明了

$$\beta^{(p-1)/q}\equiv \alpha^{(p-1)a_0/q} \pmod p$$

因此，我们从计算 $\beta^{(p-1)/q} \bmod p$ 开始。如果 $\beta^{(p-1)/q}\equiv 1 \pmod p$，那么 $a_0=0$。否则，我们连续地计算 $\gamma=\alpha^{(p-1)/q} \bmod p$，$\gamma^2 \bmod p$，…，直到对某一 i，有 $\gamma^i\equiv\beta^{(p-1)/q} \pmod p$为止。此时，我们有 $a_0=i$。

如果 $C=1$，那么我们已经求出了 $x=a_0=i$。如果 $C>1$，那么我们继续确定 a_1。确定 a_1 的步骤如下：设 $\beta_1=\beta\alpha^{-a_0}$，$x_1=\log_\alpha \beta_1 \bmod q^C$。易知，$x_1=\sum_{i=1}^{C-1} a_i q^i$。由此可得 $\beta_1^{(p-1)/q^2}\equiv \alpha^{(p-1)a_1/q} \pmod p$。所以我们来计算 $\beta_1^{(p-1)/q^2} \bmod p$，然后找到 i 使得 $\gamma^i\equiv\beta_1^{(p-1)/q^2} \pmod p$。此时，我们有 $a_1=i$。

如果 $C=2$，那么我们就得出了 $x=a_0+a_1 q$；否则，我们重复上述过程 $C-2$ 次，可获得 $a_2, a_3, \cdots, a_{C-1}$。

下面我们来描述计算离散对数的 Pohlig-Hellman 算法。在这个算法中，假定 α 是 Z_p 的一个本原元，q 是满足下列条件的一个素数：$p-1\equiv 0 \pmod{q^c}$，并且 $p-1 \not\equiv 0 \pmod{q^{c+1}}$。算法计算 $a_0, a_1, \cdots, a_{c-1}$，这里 $\log_\alpha \beta \bmod q^c = \sum_{i=0}^{c-1} a_i q^i$。

计算 $\log_\alpha \beta \bmod q^c$ 的 Pohlig-Hellman 算法如下：

(1) 计算 $\gamma_i = \alpha^{(p-1)i/q} \bmod p, 0 \leqslant i \leqslant q-1$；

(2) 置 $j=0, \beta_j = \beta$；

(3) 当 $j \leqslant c-1$ 时，完成下列各步骤：

(4) 计算 $\delta = \beta_j^{(p-1)/q^{j+1}} \bmod p$；

(5) 找到 i 使得 $\delta = \gamma_i$；

(6) $a_j = i$；

(7) $\beta_{j+1} = \beta_j \alpha^{-a_j q^j} \bmod p$；

(8) $j = j+1$。

现在我们用一个例子来图示 Pohlig-Hellman 算法。

例 6.4.3 假定 $p=29$，那么 $n=p-1=28=2^2\times 7^1$。取 $\alpha=2, \beta=18$，α 是 Z_{29} 的一个本原元，现在我们来计算 $a=\log_2 18$。

首先计算 $a \bmod 4$。设 $q=2, c=2, \gamma_0=1, \gamma_1 = \alpha^{28/2} \bmod 29 = 28$。$\delta = \beta^{28/2} \bmod 29 = 18^{14} \bmod 29 = 28$。因此，$a_0=1$。计算 $\beta_1 = \beta_0 \alpha^{-1} \bmod 29 = 9$，$\beta_1^{28/4} \bmod 29 = 9^7 \bmod 29 = 28$，因为 $\gamma_1 = 28$，所以 $a_1 = 1$。因此 $a \equiv 3 \pmod 4$。

其次计算 $a \bmod 7$。设 $q=7, c=1, \beta^{28/7} \bmod 29 = 18^4 \bmod 29 = 25, \gamma_1 = \alpha^{28/7} \bmod 29 = 2^4 \bmod 29 = 16$。通过计算可知，$\gamma_2 = 24, \gamma_3 = 7, \gamma_4 = 25$。因此，$a_0 = 4$，从而 $a \equiv 4 \pmod 7$。

最后，由中国剩余定理解同余方程组 $\begin{cases} a \equiv 3 \pmod 4 \\ a \equiv 4 \pmod 7 \end{cases}$ 可得 $a \equiv 11 \pmod{28}$，即 $\log_2 18 = 11$。

6.4.2.3 指数计算方法

本小节对计算离散对数的指数计算方法的基本思想作一简述。该方法使用了一个因子基，即小素数的一个集合 $\boldsymbol{B}$。假定 $\boldsymbol{B} = \{p_1, p_2, \cdots, p_B\}$。第一步（一个预处理步）是找到因子基中的 B 个素数的对数。第二步是利用因子基中的元素的对数的知识计算一个期望的元素 β 的对数。

在预计算中，我们构造 $C=B+10$ 个模 p 的同余式：

$$\alpha^{x_j} \equiv p_1^{a_{1j}} p_2^{a_{2j}} \cdots p_B^{a_{Bj}} \pmod p \qquad 1 \leqslant j \leqslant C$$

这些同余式等价于同余式

$$x_j \equiv a_{1j} \log_\alpha p_1 + \cdots + a_{Bj} \log_\alpha p_B \qquad 1 \leqslant j \leqslant C$$

给定 B 个未知量 $\log_\alpha p_i (1 \leqslant i \leqslant B)$ 的 C 个同余式，我们希望有唯一的一个模 $p-1$ 解。如果是这样，那么我们能计算因子基中的元素的离散对数。然而，我们如何来产生所期望的形式的同余式呢？一个基本的方法是取一个随机值 x，计算 $\alpha^x \bmod p$，然后确定是否 $\alpha^x \bmod p$ 的所有因子都在 $\boldsymbol{B}$ 之中（例如使用试除法）。

假定我们已经成功地完成了预计算步骤，现在我们利用一个 Las Vegas 型概率算法

来计算离散对数 $\log_\alpha\beta$。选择一个随机整数 $s(1\leqslant s\leqslant p-2)$并计算 $\gamma=\beta\alpha^s \bmod p$。下面我们试图在因子基 $\boldsymbol{B}$ 上分解 γ。如果 γ 可在 $\boldsymbol{B}$ 上分解，则可设

$$\beta\alpha^s \equiv p_1^{c_1}p_2^{c_2}\cdots p_B^{c_B}(\bmod\ p)$$

这个同余式等价于

$$\log_\alpha\beta + s \equiv c_1\log_\alpha p_1 + c_2\log_\alpha p_2 + \cdots + c_B\log_\alpha p_B(\bmod(p-1))$$

该式中除了 $\log_\alpha\beta$ 之外，其余值都是已知的，所以容易算出 $\log_\alpha\beta$。

人们认为在合理的假设下，该算法在预计算阶段的平均运行时间为 $O\left(e^{(1+o(1))\sqrt{\ln p\ln\ln p}}\right)$，找到单个的离散对数的平均运行时间为 $O\left(e^{(1/2+o(1))\sqrt{\ln p\ln\ln p}}\right)$。

6.4.3 离散对数的比特安全性

本小节我们来看一看关于离散对数的部分信息问题[37]。特别地，我们考虑一下一个离散对数的单个比特是容易计算的还是难计算的。首先我们介绍一下离散对数的第 i 比特问题。离散对数的第 i 比特问题可叙述为：假定 p 是一个素数，$\alpha\in Z_p^*$ 是一个本原元，$\beta\in Z_p^*$，i 是一个整数，$1\leqslant i\leqslant[\log_2(p-1)]$，计算 $L_i(\beta)$，其中 $L_i(\beta)$ 表示 $\log_\alpha\beta$ 的第 i 个比特。

下面我们来说明计算一个离散对数的最低比特是容易的，即计算 $L_1(\beta)$是容易的。

假定 α 是 Z_p 的一个本原元，p 为素数，$QR(p)$表示模 p 的二次剩余集。如果 a 是偶数时，显然 $\alpha^a\in QR(p)$。因为 $(p-1)/2$ 个元素 $\alpha^0 \bmod p, \alpha^2 \bmod p, \cdots, \alpha^{p-3} \bmod p$ 两两互不相同，并且 $|QR(p)|=\dfrac{p-1}{2}$，所以

$$QR(p)=\left\{\alpha^{2i} \bmod p \,\middle|\, 0\leqslant i\leqslant\frac{(p-3)}{2}\right\}$$

因此，$\beta\in QR(p)\Leftrightarrow\log_\alpha\beta$ 为偶数$\Leftrightarrow L_1(\beta)=0$。但是由欧拉(Euler)准则知，$\beta\in QR(p)\Leftrightarrow\beta^{(p-1)/2}\equiv 1(\bmod\ p)$。所以我们可得出如下的计算 $L_1(\beta)$的有效公式：

$$L_1(\beta)=\begin{cases}0 & \beta^{(p-1)/2}\equiv 1(\bmod\ p)\\ 1 & \text{否则}\end{cases}$$

现在我们来考虑 $L_i(\beta)(i>1)$的计算。

假定 $p-1=2^st$，t 为奇数。可证明对任意的 $i\leqslant s$，计算 $L_i(\beta)$是容易的。但计算$L_{s+1}(\beta)$在下列意义下是困难的：计算 $L_{s+1}(\beta)$的任何假想的算法(或预言)可用来找 Z_p 中的离散对数。下面我们就 $s=1$，$p\equiv 3(\bmod\ 4)$的情况来证明这一结果，即说明如何用计算 $L_2(\beta)$的任何预言计算 Z_p 中的离散对数问题。

如果 β 是 Z_p 中的一个二次剩余，$p\equiv 3(\bmod\ 4)$，那么 $\pm\beta^{\frac{p+1}{4}} \bmod p$ 是 $\beta \bmod p$ 的两个平方根，并且对任何 $\beta\neq 0$，有 $L_1(\beta)\neq L_1(p-\beta)$。这是因为，如果 β 是 $Z_p(p\equiv 3(\bmod\ 4))$中的一个二次剩余，那么可设 $\beta=y^2 \bmod p, y\in Z_p$，从而

$$\left(\pm\beta^{\frac{p+1}{4}}\right)^2\equiv\beta^{\frac{p+1}{2}}\equiv y^{p+1}\equiv y^{p-1}\cdot y^2\equiv y^2\equiv\beta(\bmod\ p)$$

又

$$\beta^{\frac{p+1}{4}}\not\equiv-\beta^{\frac{p+1}{4}}(\bmod\ p)$$

这说明 $\pm\beta^{\frac{p+1}{4}} \bmod p$ 是 $\beta \bmod p$ 的两个平方根。对任何 $\beta\neq 0$，可设 $\beta\equiv\alpha^a(\bmod\ p)$，$0\leqslant a\leqslant$

$p-2$，因为 $\alpha^{\frac{p-1}{2}} \equiv -1 \pmod{p}$，则 $\alpha^{a+(p-1)/2} \equiv -\beta \pmod{p}$。又因为 $p \equiv 3 \pmod{4}$且$(p-1)/2$是奇数，所以 $L_1(\beta) \neq L_1(p-\beta)$。

现在，假定 $\beta = \alpha^a \bmod p$，a 是某一未知偶指数，那么

$$\beta^{\frac{(p+1)}{4}} \equiv \alpha^{\frac{a}{2}} \pmod{p} \text{ 或 } -\beta^{\frac{(p+1)}{4}} \equiv \alpha^{\frac{a}{2}} \pmod{p}$$

如果我们知道值 $L_2(\beta)$，那么可根据 $L_2(\beta) = L_1(\alpha^{\frac{a}{2}})$确定这两种可能性中的哪一个成立。下面我们基于这一事实介绍一个用计算 $L_2(\beta)$的一个预言来计算 $Z_p^*(p \equiv 3 \pmod{4})$中的离散对数的算法。其具体步骤如下：

(1)$x_0 = L_1(\beta)$；

(2)$\beta = \beta / \alpha^{x_0} \bmod p$；

(3)$i = 1$

(4)While $\beta \neq 1$ do

(5)$x_i = L_2(\beta)$；

(6)$\gamma = \beta^{\frac{(p+1)}{4}} \bmod p$；

(7)if $L_1(\gamma) = x_i$ then

(8)$\beta = \gamma$；

(9)else

(10)$\beta = p - \gamma$；

(11)$\beta = \beta / \alpha^{x_i} \bmod p$；

(12)$i = i + 1$。

在算法结束时，$\{x_i\}_{i \geqslant 0}$构成了 $\log_\alpha \beta$ 的一个二进制表示，即 $\log_\alpha \beta = \sum_{i \geqslant 0} x_i 2^i$。

利用数学归纳法可对上述算法给出一个正式的证明。设 $x = \log_\alpha \beta = \sum_{i \geqslant 0} x_i 2^i$。对 $i \geqslant 0$，定义 $Y_i = \left\lfloor \frac{x}{2^{i+1}} \right\rfloor$。将 β_0 定义为该算法在第(2)步的值 β。对 $i \geqslant 1$，将 β_i 定义为该算法在当圈的第 i 次迭代期间第(11)步的值 β。递归地可证明，对所有的 $i \geqslant 0$，有 $\beta_i \equiv \alpha^{2Y_i} \pmod{p}$。易知，$2Y_i = Y_{i-1} - x_i$，从而可得出，对 $i \geqslant 0$，有 $x_{i+1} = L_2(\beta_i)$。由此可证明该算法是正确的。详细证明过程请读者自己写出。

6.5 其它公钥密码算法

前面我们比较详细地介绍了两种公钥密码算法，即 RSA 算法和 ElGamal 算法。自从公钥密码思想诞生以来，人们提出了各种各样的公钥密码算法[38]，除了上述介绍的两种算法外，还有 Rabin 算法[39]、Merkle-Hellman 背包算法[40]、Chor-Rivest 算法[41]、McEliece 算法[42]、椭圆曲线密码算法[43,44]、有限自动机算法[45]、基于身份的公钥密码算法[61,62,63]等等。本节我们再来简要介绍几种公钥密码算法。

6.5.1 Rabin 算法

Rabin 算法的安全性是基于因子分解的困难性。假定模 $n = pq$ 不能被分解，那么 Rabin 算法关于选择明文攻击是计算上安全的，下面我们来描述该算法。

设 n 是两个不同素数 p 和 q 的乘积，$p\equiv 3(\mathrm{mod}4)$，$q\equiv 3(\mathrm{mod}4)$。$\boldsymbol{P}=\boldsymbol{C}=Z_n$，令 $\boldsymbol{K}=\{(n,p,q,B)\mid 0\leqslant B\leqslant n-1)\}$。对每一个 $K=(n,p,q,B)$，定义

$$E_K(x)=x(x+B)\bmod n \qquad x\in P$$

$$D_K(y)=\sqrt{\frac{B^2}{4}+y}-\frac{B}{2} \qquad y\in C$$

值 n 和 B 是公开的，而 p 和 q 是保密的。

现在我们来说明 Rabin 算法中的加密函数 E_K 不是一个双射，因而密文可被解密成好几个明文。事实上，对任何一个给定的密文，有 4 个明文与之对应。设 w 是 1 模 n 的四个平方根之一，$x\in Z_n$，则易知

$$E_K\left(w\left(x+\frac{B}{2}\right)-\frac{B}{2}\right)=E_K(x)$$

加密成 $E_K(x)$ 的四个明文是

$$x,\ -x-B,w(x+B/2)\text{ 和 }-w\left(x+\frac{B}{2}\right)$$

这里 w 是 1 模 n 的非平凡平方根。一般地，没有办法区分这四个可能的明文中哪一个是真正的明文，除非明文包含充分的冗余度来说明删除这四个中的哪三个。

下面我们来看一下 Rabin 算法的解密过程。该算法的解密过程就是从同余方程 $x^2+Bx\equiv y(\mathrm{mod}\ n)$ 中确定出 x。

令 $x=x_1-\frac{B}{2}$，可将上述方程变为 $x_1^2\equiv\frac{B^2}{4}+y(\mathrm{mod}\ n)$。记 $C=\frac{B^2}{4}+y$，则 $x_1^2\equiv C(\mathrm{mod}\ n)$。可见，解密可归于求模 n 的平方根问题。上述方程等价于解同余式方程组

$$\begin{cases}x_1^2\equiv C(\mathrm{mod}\ p)\\ x_1^2\equiv C(\mathrm{mod}\ q)\end{cases}$$

方程 $x_1^2\equiv C(\mathrm{mod}\ p)$ 关于模 p 有两个平方根，方程 $x_1^2\equiv C(\mathrm{mod}\ q)$ 关于模 q 有两个平方根。由中国剩余定理可知，上述方程组关于模 n 有四个平方根。

当 $p\equiv 3(\mathrm{mod}\ 4)$ 时，由前面的讨论知，$\pm C^{\frac{p+1}{4}}$ 为 C mod p 的两个平方根。类似地，当 $q\equiv 3(\mathrm{mod}\ 4)$ 时，$\pm C^{\frac{(q+1)}{4}}$ 为 C mod q 的两个平方根。然后直接使用中国剩余定理可获得 C mod n 的四个平方根 x_1。一旦我们确定了 x_1 的四个可能的值，我们可从 $x=x_1-\frac{B}{2}$ 得到四个可能的明文，因此，Rabin 体制的解密公式为

$$D_K(y)=\sqrt{\frac{B^2}{4}+y}-\frac{B}{2}$$

值得注意的是，当 $p\equiv 1(\mathrm{mod}\ 4)$ 时，目前还没有找到计算二次剩余模 p 的平方根的多项式时间的确定性算法。然而，有一个多项式时间的 Las Vegas 型算法。

我们现在讨论 Rabin 算法的安全性。我们将说明 Rabin 算法的任何假想的解密算法 A 可用作分解模 n 的一个 Las Vegas 型算法的一个预言。该 Las Vegas 型算法至少以 1/2 的概率分解 n，下面我们来叙述这一算法。

给定一个解密预言，分解一个 Rabin 模的算法：

(1) 选择一个随机数 r，$1\leqslant r\leqslant n-1$；

(2) 计算 $y=(r^2-B^2/4)\bmod n$；

(3) 将 y 作为预言 A 的输入，获得一个解密结果 x；

(4) 计算 $x_1=x+\frac{B}{2}$；

(5) 如果 $x_1\equiv\pm r(\mathrm{mod}\ n)$，那么停止(分解失败)；否则 $\gcd(x_1+r,n)=p$ 或 q(分解成功)。

现在对这一算法作一些解释。

因为 $y=E_K\left(r-\frac{B}{2}\right)$，所以在第(3)步可获得一个值 x。由第(4)步并注意到 $x_1^2\equiv r^2(\mathrm{mod}n)$，我们可得出 $x_1\equiv\pm r(\mathrm{mod}\ n)$或 $x_1\equiv wr(\mathrm{mod\ n})$，$w$ 是 1 模 n 的一个非平凡平方根。在第二种情况下，我们有 $n|(x_1-r)(x_1+r)$，但 $n\nmid(x_1-r)$，$n\nmid(x_1+r)$，所以 $\gcd(x_1+r,n)=p$ 或 q。

现在我们考虑一下这个算法成功的概率。对两个非零剩余 r_1 和 r_2，定义 $r_1\sim r_2\Leftrightarrow r_1^2\equiv r_2^2(\mathrm{mod}\ n)$。易知关系"~"是一个等价关系，即"~"满足反身性($r\sim r$)、对称性(若 $r_1\sim r_2$，则 $r_2\sim r_1$)和传递性(若 $r_1\sim r_2$，$r_2\sim r_3$，则 $r_1\sim r_3$)。记 $[r]=\{r_1\in Z_n|r_1\sim r\}$，称 $[r]$ 为包含 r 的一个等价类，所有的非零等价类都包含四个元素，即 $[r]=\{\pm r,\pm wr\mathrm{mod}n\}$，这里 w 是 1 模 n 的一个非平凡平方根。在上述算法中，任何在同一等价类中的两个值 r 都将产生同样的值 y。现在考虑当给定 y 时由预言 A 返回的值 x，我们有 $[x_1]=\{\pm x_1,\pm wx_1\}$。如果 $r=\pm x_1$，则算法失败；如果有 $r=\pm wx_1$，则算法成功。因为 r 是随机选择的，所以它等可能地等于这四个可能值中的任何一个。故算法成功的概率是 1/2。

有趣的是可证明 Rabin 算法对选择明文攻击是安全的，然而该算法对选择密文攻击是完全不安全的。事实上，利用上述提出的分解一个 Rabin 模的算法就能说明 Rabin 算法对选择明文攻击是安全的。另外，也可用此算法说明选择密文攻击能破译 Rabin 算法。在选择密文攻击中，预言 A 由用户 B 的解密算法来代替。

关于 Rabin 算法的比特安全性的讨论可参阅文献[12]。

6.5.2 Merkle-Hellman 背包算法

Merkle-Hellman 背包算法的安全性是基于背包问题的困难性。所谓背包问题是指，给定一个正整数集合 $\{s_1,s_2,\cdots,s_n\}$ 和一个正整数 T，找一个 0,1 向量 $x=(x_1,x_2,\cdots,x_n)$ 使得

$$\sum_{i=1}^{n}x_is_i=T$$

记为 $(s_1,s_2,\cdots,s_n,T)$。除了文献[41]中所介绍的 Chor-Rivest 背包型算法外，所有的背包算法都是不安全的。Chor-Rivest 背包型算法虽然认为是安全的，但其计算量太大，不太实用。对 Chor-Rivest 背包算法感兴趣的读者请参阅文献[41]，Merkle-Hellman 背包算法于 80 年代早期[46,47]就被破译。随后人们给出了一系列的变形，这些变形也逐渐被破译，关于这些变形及其分析可参阅文献[48,49,50,51]。本小节我们来介绍 Merkle-Hellman 背包算法，虽然它被破译，但它的设计思想及其破译方法给人们认识公钥密码以诸多启示。

背包问题是一个 NP 完全问题。这意味着没有多项式时间算法来解决这一问题，但这并不意味着没有多项式时间算法解决其某些特例。下面我们给出解决背包问题的一个实例的多项式时间算法。

如果正整数序列 $s_1,s_2,\cdots,s_n$ 满足条件

$$\sum_{i=1}^{k-1} s_i < s_k \qquad 2 \leqslant k \leqslant n$$

我们称序列 $s_1, s_2, \cdots, s_n$ 为超递增序列，记为 $s=(s_1, s_2, \cdots, s_n)$。相应的背包问题称为超递增背包问题(superincreasing knapsack problem)。超递增背包问题可在时间 $O(n)$ 内很容易地解决，如果有解，解一定是唯一的。解决一个超递增背包问题的算法如下：

(1) for $i=n$ downto 1 do
(2) if $T \geqslant s_i$ then
(3) $T=T-s_i$
(4) $x_i=1$
(5) else
(6) $x_i=0$
(7) if $\sum_{i=1}^{n} x_i s_i = T$ then
(8) $x=(x_1, x_2, \cdots, x_n)$是解
(9) else
(10) 没有解。

假定 $s=(s_1, s_2, \cdots, s_n)$是超递增的，考虑函数

$$E_s: \{0,1\}^n \to \left\{0, 1, 2, \cdots, \sum_{i=1}^{n} s_i\right\}$$

定义加密变换为 $E_s(x_1, x_2, \cdots, x_n)=\sum_{i=1}^{n} x_i s_i$。因为 s 是超递增的，所以 E_s 是一个双射。上述解超递增背包问题的算法是相应的解密算法。这样的一个系统是完全不安全的，因为任何人都可以解密用这种方式加密的消息。因此，Merkle 和 Hellman 采取了如下的策略：应用一个所谓的模变换将一个超递增背包变换成一个在没有辅助信息下难于求解的陷门背包。

首先选定一个超递增序列 $s=(s_1, s_2, \cdots, s_n)$。其次选择一个素数模 $p\left(p > \sum_{i=1}^{n} s_i\right)$ 和一个乘数 $a(1 \leqslant a \leqslant p-1)$作模变换：

$$t_i = as_i \bmod p \qquad 1 \leqslant i \leqslant n$$

记 $t=(t_1, t_2, \cdots, t_n)$，设

$$\boldsymbol{P} = \{0,1\}^n, \boldsymbol{C} = \{0, 1, \cdots, n(p-1)\} \qquad \boldsymbol{K} = \{(s, p, a; t)\}$$

公开 t，保密 s，p 和 a。

对 $K=(s, p, a, t)$，定义

$$E_K(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{n} x_i t_i \qquad (x_1, x_2, \cdots, x_n) \in \boldsymbol{P}$$

对 $0 \leqslant y \leqslant n(p-1)$，定义 $Z = a^{-1} y \bmod p$，解超递增背包问题 $(s_1 s_2, \cdots, s_n, Z)$，获得 $D_K(y)=(x_1, x_2, \cdots, x_n)$。

下面我们用一个例子来图示 Merkle-Hellman 算法的加解密过程。

例 6.5.1 假定 $s=(1,3,5,10)$，$p=23$，$a=3$，$a^{-1} \bmod 23=8$，则公开的背包向量 $t=(3,9,15,7)$。

现在如果 A 想加密明文 $x=(1,0,0,1)$，A 计算 $y=3\times1+7\times1=10$。如果 B 收到密文 $y=10$，B 首先计算 $Z=a^{-1}y\bmod p=11$，然后解超递增背包$(1,3,5,10;11)$，可得明文 $x=(1,0,0,1)$。

6.5.3 McEliece 算法

McEliece 算法是基于代数编码理论的一种公钥密码算法，它的安全性是基于译一般线性纠错码的困难性。目前为止，仍然认为这种算法是安全的，但该算法存在着一些缺陷，诸如公开密钥庞大，数据扩展大。McEliece 算法的设计思想与 Merkle-Hellman 算法的设计思想相似。译一般的线性码是一个 NP-完全问题，但一些码诸如 Goppa 码有多项式时间的译码算法。McEliece 的策略是将一个有多项式时间译码算法的 Goppa 码伪装成一个在没有辅助信息下难译的线性码。

首先我们介绍一点所需要的编码理论知识。

设 k,n 是两个正整数，$k<n$。一个二元$[n,k]$线性码 C 是 F_2^n 的一个 k 维线性子空间，n 称为码长，k 为码的维数即信息位的长度。C 中的向量称为码字。码字 u 的非零分量的个数称为 u 的汉明重量，记为 $W_H(u)$。称 $d_H(u,v)=W_H(u-v)$为码字 u,v 之间的汉明距离，码 C 的最小距离定义为

$$d=\min_{\substack{u,v\in C\\ u\neq v}} d_H(u,v)$$

如果 $d\geqslant 2t+1$，则发送 C 的任一码字 u，u 在传输时发生的错误不多于 t 时，都可以被接收端找出并予以纠正，这是因为 u 是 C 中距离接收向量 $u'=u+e$ 最近的唯一码字，e 称为错误向量，$W_H(e)\leqslant t$。

$[n,k]$线性码 C 中 k 个线性无关的码字向量构成码 C 的一个基，它们给出的 $k\times n$ 矩阵 G 称为码 C 的一个生成矩阵(generator matrix)。C 的正交补空间 $C^{\perp}$是一个$[n,n-k]$线性码，其生成矩阵 H 称为码 C 的一个一致校验矩阵(parity check matrix)。显然有 $GH'=0$。由线性代数易知，假定码 C 的生成矩阵和校验矩阵分别为 G 和 H，则 $x\in F_2^n$ 是一个码字，即 $x\in C\Leftrightarrow xH'=0$，其中 H' 表示 H 的转置。这说明

$$C=\{x\in F_2^n \mid xH'=0\}$$

设 $g(Z)$是有限域 F_{2^m}上的一个 t 次不可约多项式，$t>1$，$n=2^m$。F_{2^m}上的全部元素记为 α_i，$1\leqslant i\leqslant n$。

令

$$H'=\begin{pmatrix}1 & 1 & \cdots & 1\\ \alpha_1 & \alpha_2 & \cdots & \alpha_n\\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2\\ \vdots & \vdots & \cdots & \vdots\\ \alpha_1^{t-1} & \alpha_2^{t-1} & \cdots & \alpha_n^{t-1}\end{pmatrix}\begin{pmatrix}g^{-1}(\alpha_1) & 0 & \cdots & 0\\ 0 & g^{-1}(\alpha_2) & \cdots & 0\\ \vdots & 0 & g^{-1}(\alpha_3) & \vdots\\ \vdots & \cdots & \ddots & \vdots\\ 0 & \cdots & 0 & g^{-1}(\alpha_n)\end{pmatrix}$$

将 H'中的元素用相应的 m 长二元列向量取代，得出 F_2 上的$(mt)\times n$ 矩阵记为 H。以 H 为校验矩阵的码

$$\Gamma=\{u\in F_2^n \mid uH'=0\}$$

称为二元不可约 Goppa 码，称 $g(Z)$ 为 Goppa 多项式。通常将生成矩阵为 G 的 Goppa 码记为 $\Gamma(G)$。关于 Goppa 码有以下结论，其详细证明参见文献[52]。

(1)$\Gamma(G)$的维数 $k\geqslant n-mt$，最小距离 $d\geqslant 2t+1$；

(2)存在时间复杂度为 $O(nt)$的快速纠错译码算法；

(3)F_{2^m}上的 t 次不可约多项式的数目为 $O(2^{mt}/t)$，这表明，对给定的 m,t，用随机取法能以 $1/t$ 的概率获得 t 次不可约多项式，其不可约性有快速算法加以测试，参见文献[53]。

下面我们来描述 McEliece 算法。

设 G 是一个$[n,k,d]$Goppa 码 C 的一个生成矩阵，$n=2^m$，$d=2t+1$，$k=n-mt$。设 S 是 F_2 上的一个 $k\times k$ 阶可逆矩阵，P 是 F_2 上的一个 $n\times n$ 阶置换矩阵，令 $G'=SGP$，$\boldsymbol{P}=F_2^k$，$\boldsymbol{C}=F_2^n$，$\boldsymbol{K}=\{(G,S,P,G')\}$，这里的 G,S,P,G' 如上所述。公开 G'，保密 G,S 和 P。

对 $K=(G,S,P,G')$，定义 $E_K(x,e)=xG'+e$，这里 $e\in F_2^n$ 是一个重量不超过 t 的随机向量。

解密过程如下：

(1) 计算 $y_1=yP^{-1}$；

(2) 译码字 y_1，获得 $y_1=x_1+e_1$，$x_1\in C$；

(3) 计算 $x_0\in F_2^k$，使得 $x_0G=x_1$；

(4) 计算 $x=x_0S^{-1}$。

在该算法的实际实现中，McEliece 建议取如下参数：$m=10$，$t=50$，此时 $n=2^{10}=1024$，$k=n-mt=524$，$d=101$，Goppa 码 C 是一个[1024,524,101]线性码。明文的长度为 524 比特，密文的长度为 1024 比特。公钥是一个 524×1024 阶的二元矩阵。

6.5.4 二次剩余算法(概率加密)

二次剩余(quadratic residue，简记为 QR)算法作为一种公钥密码算法，它与一般的公钥密码算法不同，使用了所谓的概率加密(probabilitic encryption)技术。概率加密是 Goldwasser 和 Micali 的观点[54]。其基本目的是使敌手在多项式时间内不能从密文获得有关明文的任何信息。这个目的可通过在公钥密码算法中使用概率加密而不是确定性加密来实现。现在我们来介绍一种概率加密算法——二次剩余算法[54]。

设 $n=pq$，p 和 q 是素数。选择一个正整数 $t\notin QR(n)$且$\left(\frac{t}{n}\right)=1$，$QR(n)$表示模 n 的二次剩余之集，$\left(\frac{t}{n}\right)$表示 t 关于模 n 的 Jacobi 符号。公开 n,t，保密 p 和 q。$\boldsymbol{P}=\boldsymbol{C}=Z_2^n$

$$\boldsymbol{K}=\left\{(n,t,p,q)\mid n=pq,p,q\text{ 为素数},t\notin QR(n),\left(\frac{t}{n}\right)=1\right\}$$

对 $K=(n,t,p,q)$，定义加密变换为 $E_K(x,r)=y=(y_1,y_2,\cdots,y_n)$，其中 $y_i=t^{x_i}r_i^2 \bmod n$，$1\leqslant i\leqslant n$，$x=(x_1,x_2,\cdots,x_n)\in\boldsymbol{P}$，$r=(r_1,r_2,\cdots,r_n)$是随机选择的一个向量。

解密变换为 $D_K(y)=(x_1,x_2,\cdots,x_n)$，其中

$$x_i=\begin{cases}0 & y_i\in QR(n)\\ 1 & y_i\notin QR(n)\end{cases}\qquad 1\leqslant i\leqslant n\qquad y=(y_1,y_2,\cdots,y_n)\in\boldsymbol{C}$$

在二次剩余算法中，知道 n 的分解的用户可通过下列方法确定 $y_i\in QR(n)$还是 $y_i\notin QR(n)$：

(1) 计算$\left(\frac{y_i}{p}\right)=(-1)^{(p-1)/2} \bmod p$，$\left(\frac{y_i}{q}\right)=(-1)^{(q-1)/2} \bmod q$；

(2) $y_i \in QR(n) \Leftrightarrow \left(\frac{y_i}{p}\right)=1, \left(\frac{y_i}{q}\right)=1$。

关于概率加密算法的进一步讨论可参阅文献[8,54,55,56]。

6.5.5 椭圆曲线密码算法

目前已有大量的文献讨论了椭圆曲线密码算法，关于椭圆曲线算法的研究可参阅文献[57]。椭圆曲线密码算法之所以引起人们的广泛关注，是因为它除了理论上的意义之外，有以下两个明显的优点：

(1) 具有短的密钥长度，这意味着小的带宽和存贮要求，这些因素在某些应用场合十分有用，诸如在智能卡系统中；

(2) 所有的用户可选择同一基域 F 上的不同的椭圆曲线 E，这可使所有的用户使用同样的硬件完成域算术，为保证安全性，只有椭圆曲线 E 选择的不同。

本小节我们来介绍一个椭圆曲线密码算法，称作 Menezes-Vanstone 椭圆曲线算法[58]。在介绍该算法之前，我们先对椭圆曲线中的一些基本概念作一介绍。由于椭圆曲线算法的数学原理很复杂，这里只能作一简单的介绍。对椭圆曲线算法感兴趣的读者请参阅文献[57,59,60]。

设 $p>3$ 是一个素数，在 Z_p 上的椭圆曲线 $y^2=x^3+ax+b$ 是同余式 $y^2 \equiv x^3+ax+b \pmod p$ 的解 $(x,y) \in Z_p \times Z_p$ 的集合，连同一个称作无穷远点的特殊点 O。这里 $a,b \in Z_p$，a 和 b 满足条件 $4a^3+27b^2 \not\equiv 0 \pmod p$。

在一个椭圆曲线 E 上定义适当的运算后可形成一个阿贝尔群，该运算称作加法运算，记作“+”。定义如下：假定 $P=(x_1,y_1)$，$Q=(x_2,y_2) \in E$。如果 $x_2=x_1$，$y_2=-y_1$，则 $P+Q=O$；否则，$P+Q=(x_3,y_3)$，这里 $x_3=\lambda^2-x_1-x_2$，$y_3=\lambda(x_1-x_3)-y_1$，

$$\lambda=\begin{cases}\dfrac{y_2-y_1}{x_2-x_1} & P \neq Q \\ \dfrac{3x_1^2+a}{2y_1} & P=Q\end{cases}$$

对所有的 $P \in E$，定义 $P+O=O+P=P$。

E 在上述定义的运算下形成一个 Abelian 群，单位元为 O，大多数规则可直接验证，但证明结合律是比较难的。逆是很容易计算的，$(x,y) \in E$ 的逆为 $(x,-y)$，写作 $-(x,y)$。

定义在 Z_p（p 素数，$p>3$）上的一个椭圆曲线 E 大约有 p 个点。著名的 Hasse 定理告诉我们，E 上的点的个数（记为 $\#E$）满足下列的不等式：

$$p+1-2\sqrt{p} \leqslant \#E \leqslant p+1+2\sqrt{p}$$

计算 $\#E$ 的精确值是比较难的，但 Schoof 算法可在时间 $O((\log_2 p)^8)$ 内计算出 $\#E$ 的精确值。现在假定我们能计算 $\#E$，我们想找到 E 的一个循环子群使得在这个子群上的离散对数问题是难处理的。所以我们需要知道 E 的结构，下面的定理对 E 的结构给出了大量的信息。

定理 6.5.1 设 $p>3$，p 是一个素数，E 是定义在 Z_p 上的一个椭圆曲线，则存在正整数 n_1 和 n_2，使得 E 同构于 $Z_{n_1} \times Z_{n_2}$，而且 $n_2 | n_1$，$n_2 | (p-1)$。

现在我们来描述 Menezes-Vanstone 椭圆曲线密码算法。

设 E 是 $Z_p(p>3$ 是素数)上的满足下列条件的一个椭圆曲线：E 包含一个在其上离散对数问题是难处理的循环子群 H。$\boldsymbol{P}=Z_p^* \times Z_p^*$，$\boldsymbol{C}=E\times Z_p^* \times Z_p^*$，$\boldsymbol{K}=\{(E,\alpha,a,\beta)\mid \beta=a\alpha,\alpha\in E\}$。公开 α 和 β，保密 a。

对 $K=(E,\alpha,a,\beta)$，对一个秘密随机数 $k\in Z_{|H|}$ 和 $x=(x_1,x_2)\in Z_p^* \times Z_p^*$，定义 $E_K(x,k)=(y_0,y_1,y_2)$，这里 $y_0=k\alpha$，$(c_1,c_2)=k\beta$，

$$y_1 = c_1x_1 \bmod p,\ y_2 = c_2x_2 \bmod p$$

对一个密文 $y=(y_0,y_1,y_2)$，定义 $D_K(y)=(y_1c_1^{-1} \bmod p, y_2c_2^{-1} \bmod p)$，这里 $ay_0=(c_1,c_2)$。

例 6.5.2 设 E 是在 Z_{11} 上的椭圆曲线 $y^2=x^3+x+6$。E 上有 13 个点，即 O，(2,7)，(2,4)，(10,2)，(7,2)，(8,8)，(5,2)，(3,6)，(3,5)，(5,9)，(8,3)，(7,9)，(10,9)。因为任何素数阶群都是循环的，所以 E 同构于 Z_{13}，并且除了无穷远点 O 外，任何其它点都是生成元(本原元)。假定 $\alpha=(2,7)$，用户 B 的秘密指数 $a=2$，则 $\beta=2\alpha$。因为 $2\alpha=\alpha+\alpha$，我们首先计算 $\lambda=(3\times 2^2+1)(2\times 7)^{-1} \bmod 11=2\times 3^{-1} \bmod 11=8$，这样我们有 $x_3=(8^2-2-2) \bmod 11=5$，$y_3=(8(2-5)-7) \bmod 11=2$。所以，$\beta=2\alpha=(5,2)$。

假定 A 想加密明文 $x=(x_1,x_2)=(9,1)$，注意明文 x 未必是 E 上的点，这里的 x 就不是 E 上的点。A 选择一个随机值 $k=3$。A 首先计算 $y_0=k\alpha=3\alpha$ 和 $k\beta$。由 E 上的运算规则可计算出 $y_0=3\alpha=(8,3)$，$k\beta=3(5,2)=(7,9)$。所以，$c_1=7$，$c_2=9$。其次，A 计算 $y_1=c_1x_1 \bmod p=7\times 9 \bmod 11=8$ 和 $y_2=c_2x_2 \bmod p=9\times 1 \bmod 11=9$。最后，A 将密文 $y=(y_0,y_1,y_2)=((8,3),8,9)$发送给 B。

当 B 收到密文 y 时，B 首先计算 $(c_1,c_2)=ay_0=2(8,3)=(7,9)$。然后计算 $x=(y_1c_1^{-1} \bmod p, y_2c_2^{-1} \bmod p)=(8\times 7^{-1} \bmod 11, 9\times 9^{-1} \bmod 11)=(9,1)$，即解出明文 x。

6.6 注记和文献

本章主要介绍了一些有代表性的公钥密码算法，特别是对两类重要的公钥算法即 RSA 算法和 ElGamal 算法作了比较详细的讨论。另外，还有些比较重要的算法，诸如 Chor-Rivest 算法[41]等，限于篇幅，文中没有介绍，感兴趣的读者请参阅有关文献。在文献[65,67]中还可以找到一些其它公钥体制的线索。

除了陶和陈的公钥算法[45,63]外，我国学者还自行设计了一些别的公钥算法，感兴趣的读者可查看中国密码学会议论文集及国内一些有关杂志。

对椭圆曲线公钥算法感兴趣的读者请阅文献[57]。文献[38]是一篇关于公钥密码学的很好的综述。

参 考 文 献

[1] Diffie, W., Hellman, M., New Directions in Cryptography, IEEE Trans. On Info. Theory Vol. IT-22(6), pp. 644—654, Nov. 1976.

[2] Rivest, R. L., Shamir, A. and Adleman, L., A Method for Obtaining Digital Signatures and Public-key Cryptosys

tem, Comm. ACM Vol. 21(2), pp. 120—126, Feb. 1978.

[3] Lempel, A., Cryptology in Transition, Computing Surveys Vol. 11(4), pp. 285—303, Dec. 1979.

[4] Shamir, A., On the Cryptocomplexity of Knapsack Systems, Proc. 11th Annual ACM Symp. On the Theory of Computing, pp. 118—129, May, 1979.

[5] Denning, E. E. R., Cryptography and Data Security, Addison-Wesley Publishing Company, 1982.

[6] Brichell, E. F., Surrey of Hardware Implementations of RSA, Advances in Cryptology-Crypto'89, Berlin: Springer-Verlag, 1990, pp. 368—370.

[7] Delaurentis, J. M., Afurther Weakness in the Common Modulus Protocol for the RSA Cryptosystem, Cryptologia, 8(1984), pp. 253—259.

[8] Stinson, D. R., Cryptography-theory and Practice, CRC Press, 1995.

[9] Hastad, J., On Using RSA with Low Exponent in a Public key Network, Advances in Cryptology-Crypto'85, Berlin, Springer-verlag, 1986, pp. 403—408.

[10] Weiner, M. J., Cryptanalysis of Short RSA Secret Exponents, IEEE Transactions on Information Theory, Vol. 36, No. 3, May 1990, pp. 553—558.

[11] Goldwasser, S., Micali, S. and Tong, P., Why and How to Establish a Common Code on a Public Network. In 23rd Annual Symposium on the Foundations of Computer Science, pp. 134—144, IEEE Press, 1982.

[12] Chor, B. Z., Two Issues in Public-key Cryptography-RSA Bit Security and a New Knapsack Type System, MIT Press, 1986.

[13] Solovay, R. and Strassen, V., A Fast Monte Carlo Test for Primality, SIAM Journal on Computing, SIAM Journal on Computing, 6(1977), pp. 84—85.

[14] Beauchemin, P., Brassard, G., Crepeau, C., Goutier, C. and Pomenance, C., The Generation of Random Numbers That Are Probably Prime, Journal of Cryptology, 1(1988), pp. 53—64.

[15] Miller, G. L., Riemann's Hypothesis and Tests for Primality, Journal of Computer and Systems Science, 13 (1976), pp. 300—317.

[16] Rabin, M. O., Probabilistic Algorithms for Testing Primality, Journal of Number Theory, 12(1980), pp. 128—138.

[17] Pomerance, C., Recent Developments in Primality Testing, Mathematics Intelligence, Vol. 3, 1981, pp. 97—105.

[18] Bond, D. J., Practical Primality Testing, Proceedings of IEEE International Conference on Secure Communications Systems, 22—23, Feb. 1984, pp. 50—53.

[19] Menezes, A. J., Van Oorschot, P. and Vanstone, S., Handbook of Applied Cryptography, CRC Press, 1996.

[20] Williams, H. C., An Overview of Factoring, Advances in Cryptology-Crypto'83, Plenum Press, 1984, pp. 71—80.

[21] Lenstra, A. K., Lenstra, Jr., H. W., Manasse, M. S. and Pollard, J. M, The Number Field Sieve, Proceedings of the 22nd ACM Symposium on the Theory of Computing, 1990, pp. 564—572.

[22] Bressoud, D. M., Factoriztion and Primality Testing, Springer-Verlag, 1990.

[23] Pemerance, C., The Quadratic Sieve Factoring Algorithm, Advances in Cryptology-Eurocrypt'84, Berlin, Springer-Verlag, 1985, pp. 169—182.

[24] Pomerance, C., Smith J. W. and Tuler, R., A Pipe-line Architecture for Factoring Large Integers with the Quadratic Sieve Algorithm, SIAM Journal on Computing, Vol. 17, No. 2, Apr. 1988, pp. 387—403.

[25] Siverman, R. D. The Multiple Polynomial Quadratic Sieve, Mathematics of Computation, Vol. 48, No. 177, Jan 1987, pp. 329—339.

[26] Lenstra, A. K., Lenstra, Jr, H. W., Manasse, M. S and Pollard, J. M., The Factorization of the Ninth Fermat Number, Mathematics of Computation, Vol. 61, No. 203, Jul. 1993, pp. 319—350.

[27] Lenstra, A. K. and Manasse, M. S., Factoring by Electronic Mail, Advances in Cryptology-Eurocrypt'89, Berlin, Springer-Verlag, 1990, pp. 355—371.

[28] Lenstra, Jr., H. W., Elliptic Curves and Number-theoretic Algorithms, Reprot 86-19, Mathematisch Instituut, Universiteit Van Amsterdam, 1986.

[29] Montgomery, P., Speeding the Pollard and Elliptic Curve Methods of Factorization, Mathematics of Computation, Vol. 48, No. 177, Jan 1987, pp. 243—264.

[30] Knuth, D., The Art of Computer Programming; Volume 2, Seminumberical Algorithms, 2nd Edition, Addison-Wesley, 1981, pp. 370—380.

[31] Morrison, M. N. and Brillhart, J., A Method of Factoring and the Factorization of F_7, Mathematics of Computation, Vol. 29, No. 129, Jan 1975, pp. 183—205.

[32] Poet, R., The Defign of Special Purpose Hardware to Factor Large Integers, Computer Physics Communications, Vol. 37, 1985, pp. 337—341.

[33] ElGamal, L., A Public Key Cryptosystem and a Signature Scheme Base on Discrete Logarithm, IEEE Transactions on Information Theory, 31(1985), pp. 469—472.

[34] Lamacchia, B. A. and Odlyzko, A. M., Computation of Discrete Logarithms in Prime Field, Designs, Codes and Cryptography, Vol. 1, 1991, pp. 46—62.

[35] Odlyzko, A., Discrete Logarithms in Finite Fields and Their Cryptographic Significance, Advances in Cryptology-Eurocrypt'84, Berlin, Springer-Verlag, 1985, pp. 224—314.

[36] Pohlig, S. C. and Hellman, M. E., An Improved Algorithm for Computing Logarithms in $GF_{(p)}$ and Its Cryptographic Significance, IEEE Transactions on Information Theory, Vol. 24, No. 1, Jan. 1978, pp. 106—111.

[37] Peralta, R., Simultaneous Security of Bits in the Discrete Log, Advances in Cryptology-Eurocrypt'85, Springer-Verlag, 1986, pp. 62—72.

[38] Diffie, W., The First Ten Years of Public-key Cryptography, in Contemporary Cryptology, the Science of Information Integrity, pp. 135—175, IEEE Press, 1992.

[39] Rabin, M. O., Digitalized Signatures and Public-key Functions as Intractable as Factorization MIT/LCS/TR-212, MIT Lab. for Computer Science, Cambridge, Mass. (Jan. 1979).

[40] Merkle, R. C. and Hellman, M. E., Hinding Information and Signatures in Trapdoor Knapsacks, IEEE Transactions on Information Theory, 24(1978), pp. 525—530.

[41] Chor, B. And Rivest, R. L., A Knapsack Type Public Key Cryptosystem Based on Arithmetic in Finite Fields, Advances in Cryptology-Eruocrypt'84, Springer-Verlag, 1985, pp. 54—65.

[42] McEliece, R. A Public-key Cryptosystem Based on Algebraic Coding Theory, DSN Progress Report, 42—44 (1978), pp. 114—116.

[43] Koblitz, N., Elliptic Curve Cryptosystems, Mathematics of Computation, 48(1987), pp. 203—209.

[44] Miller, V., Uses of Elliptic Curves in Cryptography, Advances in Cryptology-Crypto'85, Springer-Verlag, 1986, pp. 417—426.

[45] 陶仁骥、陈世华，一种有限自动机公开钥密码体制和数字签名，计算机学报，1985，No. 8, pp. 401—409.

[46] Shamir, A., A Polynomial-time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem, IEEE Transactions on Information Theory, 30(1984), pp. 699—704.

[47] Lagarias, J. C. and Odlyzko, A. M., Solving Low Density Subset Sum Problems, J. Assoc. Comp. Mach., Vol. 32, 1985, pp. 229—246.

[48] Brickell, E. F. and Odlyzko, A. M., Cryptanalysis, a Survey of Recent Results, In Contemporary Cryptology, The Science of Information Integrity, pp. 501—540, IEEE Press, 1992.

[49] Niemi, V., A New Trapdoor in Knapsacks, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 405—411.

[50] Chee, T. M., The Cryptanalysis of a New Public-key Cryptosystem Based on Modular Knapsacks, Advances in Cryptology-Crypto'91, Springer-Verlag, 1992, pp. 204—212.

[51] Joux, A. and Stern, J., Cryptanalysis of Another Knapsack Cryptosystem, Advances in Cryptology-Asiacrypt'91, Springer-Verlag, 1993, pp. 470—476.

[52] MacWilliams, F. J. and Sloane, N. J. A., The Theory of Error-Correcting Codes, North-Holland Publishing Company, 1977.

[53] Berlekamp, E. R., Algebraic Coding Theory, McGraw-Hill, New York, 1968.

[54] Goldwasser, S. and Micali, A., Probabilistic Encryption, Journal of Computer and Systems Science, 28(1984), pp. 270—299.

[55] Blum, M. and Goldwasser, S., An Efficient Probabilistic Public-key Encryption Scheme Which Hides all Partial Information, Advances in Cryptology-Crypto'84, Springer-Verlag, 1985, pp. 289—299.

[56] Vazirani, U. V. and Vazirani, V. V., Efficient and Secure Pseudo-random Number Generation, Advances in Cryptology-Crypto'84, Springer-Verlag, 1985, pp. 193—202.

[57] Menezes, A. J., Elliptic Curve Public key Cryptosystems, Kluwer Academic Publishers, 1993.

[58] Menezes, A. J. and Vanstone, S. A., Elliptic Curve Cryptosystems and Their Implementation, Journal of Cryptology, 6(1993), pp. 209—224.

[59] Bender, A. and Castagnoli, G., On the Implementation of Elliptic Cruve Cryptosystems, Advances, in Cryptology-Crypto'89, Springer-Verlag, 1990, pp. 186—192.

[60] Koblitz, N., Hyperelliptic Cryptosystems, Journal of Cryptology, Vol. 1, No. 3, 1989, pp. 129—150.

[61] Shaimr, A., Identity-based Cryptosystems and Signature, Schemes-advances in Cryptology-Crypot'84, Springer-Verlag, 1985, pp. 47—53.

[62] Tanaka, H., A *Realization* Scheme for the Identity-based Cryptosystem, Advances in Cryptology-Crypto'87, Springer-Verlag, 1988, pp. 340—349.

[63] 陶仁骐、陈世华，基于身份的密码体制和数字签名的有限自动机公开钥密码实现，密码学进展-Chinacrypt'92，科学出版社，北京，87—104 页.

[64] 戴宗铎、叶顶峰，非线性有限自动机的代数理论——兼谈 FAPK3 公钥密码体制，通信保密，No. 2，1996，45—51 页.

[65] Schneier, B., Applied Cryptography-Protocols, Algorithms, and Source Code in C, John Wiley & Sons, Inc., 1993.

[66] Fiat, A., Batch RSA, J. of Cryptology, Vol. 10, No. 2, 1997, pp. 75—88.

[67] 曹珍富，公钥密码学，黑龙江教育出版社，哈尔滨，1993.

第7章　数字签名方案

政治、军事、外交等活动中签署文件，商业上签定契约和合同，以及日常生活中在书信、从银行中取款等事务中的签字，传统上都采用手写签名或印鉴。签名起到认证、核准和生效作用。随着信息时代的来临，人们希望通过数字通信网进行迅速的、远距离的贸易合同的签名，因而数字或电子签名法应运而生，并开始用于商业通信系统，诸如电子邮件、电子转帐、办公室自动化等系统。

手写签名与数字签名的主要差别在于：

(1)签署文件方面。一个手写签名是所签文件的物理部分，而一个数字签名并不是所签文件的物理部分，所以所使用的数字签名方案必须设法把签名"绑"到所签文件上。

(2)验证方面。一个手写签名是通过和一个真实的手写签名相比较来验证的。当然，这种方法不是很安全的一种方法，相对而言是比较容易伪造某些人的手写签名。而数字签名能通过一个公开的验证算法来验证，这样，"任何人"能验证一个数字签名，安全的数字签名方案的使用将阻止伪造签名的可能性。

(3)"拷贝"方面。一个手写签名不易拷贝，因为一个文件的手写签名的拷贝通常容易与原文件区别开来。而一个数字签名容易拷贝，因为一个文件的数字签名的拷贝与原文件一样。这个特点要求我们必须阻止一个数字签名消息的重复使用，一般通过要求消息本身包含诸如日期等信息来达到阻止重复使用签名的目的。

一个签名方案至少应满足以下三个条件：(1)签名者事后不能否认自己的签名；(2)接收者能验证签名，而任何其他人都不能伪造签名；(3)当双方关于签名的真伪发生争执时，一个法官或第三方能解决双方之间发生的争执。手写签名基本上符合以上三个条件。数字签名是签以电子形式存贮的消息的一种方法，一个签名消息可以在一个通信网络中传输。基于公钥密码体制和私钥密码体制都可以获得数字签名。特别是公钥密码体制的诞生为数字签名的研究和应用开辟了一条广阔的道路。目前关于数字签名的研究主要集中于基于公钥密码体制的数字签名的研究。

根据接收者验证签名的方式可将数字签名分为真数字签名(true digital signature)和仲裁数字签名(arbitrated digital signature)两大类[1]。在真数字签名中，签名者直接把消息发送给接收者，接收者无需求助于第三方就能验证签名。而在仲裁数字签名中，签名者把签名消息经由被称作仲裁者的可信任的第三方发送给接收者，接收者不能直接地验证签名，签名的合法性是通过仲裁者作为媒介来保证，也就是说接收者要验证签名必须与仲裁者合作。

从计算能力上来分，可将数字签名分为无条件安全的数字签名和计算上安全的数字签名。现有的数字签名大部分都是计算上安全的，诸如 RSA 数字签名[2]、ElGamal 数字签名[3]等等。所谓计算上安全的数字签名是指任何伪造者伪造签名者的签名是计算上不可行的。文献[4]提出了第一个无条件安全的数字签名，这种无条件安全的数字签名的签名者和接收者都是无条件安全的。在理论上，这种数字签名在许多应用中能代替计算上安全

的数字签名，但在实际应用中是太不有效而不能被应用，这是因为在这种数字签名中需要一个复杂的交互密钥生成协议，而且签名很长。像公钥密码算法的情况一样，我们的主要目的还是设计计算上安全的数字签名。

从签名者在一个数字签名方案中所能签的消息的数来分，可将数字签名分为一次数字签名[5]和非一次数字签名。诸如 Lamport 数字签名，就是一个一次数字签名，而 RSA 数字签名是一个非一次数字签名。一次数字签名方案只能签一个消息，若签两个或两个以上不同的消息，伪造者就能伪造签名。一次数字签名方案类似于一次一密密码方案的情况，它往往具有很强的安全性。

根据数字签名方案中的验证方程是否为隐式或显式，可将数字签名分为隐式数字签名和显式数字签名[1]。现有的数字签名方案大部分为显式数字签名。也可根据数字签名的功能将数字签名分为普通数字签名和具有特殊性质的数字签名等等[6]。

数字签名方案的分类是相对而言的，并非一定要把某一数字签名方案归于某一类，上述介绍只是想让读者对现有的数字签名有一个总体印象。本章我们没有按照某一种分类去介绍数字签名方案，而是介绍一些有代表性的数字签名方案。

7.1 RSA 数字签名方案和加密

7.1.1 RSA 数字签名方案

一般地，一个数字签名方案主要由两个算法，即签名算法和验证算法组成。签名者能使用一个(秘密)签名算法 $\mathrm{Sig}(\cdot)$签一个消息 x，导致的签名 $\mathrm{Sig}(x)$可通过一个公开的验证算法$\mathrm{Ver}(\cdot,\cdot)$来验证。给定一对$(x,y)$，验证算法根据签名是否真实来作出一个“真”或“假”的回答。一个数字签名方案可由满足下列条件的五重组(P,A,K,S,V)来描述：

(1)P 是所有可能消息组成的一个有限集合；

(2)A 是所有可能签名组成的一个有限集合；

(3)K 是所有可能密钥组成的一个有限集合，即密钥空间；

(4)对每一个 $k\in K$，有一个签名算法 $\mathrm{Sig}_k(\cdot)\in S$ 和一个对应的验证算法 $\mathrm{Ver}_k(\cdot,\cdot)\in V$。每一个 $\mathrm{Sig}_k(\cdot):P\to A$ 和 $\mathrm{Ver}_k(\cdot,\cdot):P\times A\to\{$真，假$\}$是满足下列等式的函数：对每一个消息 $x\in P$ 和每一个签名 $y\in A$，有 $\mathrm{Ver}_k(x,y)=$真，当且仅当 $y=\mathrm{Sig}_k(x)$。对每一个 $k\in K$，$\mathrm{Sig}_k(\cdot)$和 $\mathrm{Ver}_k(\cdot,\cdot)$都是多项式时间函数，$\mathrm{Ver}_K(\cdot,\cdot)$是一个公开函数，而 $\mathrm{Sig}_k(\cdot)$是一个秘密函数。

现在我们基于 RSA 公钥密码算法来建立一个数字签名方案，称作 RSA 数字签名方案或算法。

设 $n=pq$，p 和 q 是两个素数，$\mathbf{P}=\mathbf{A}=Z_n$，定义 $\boldsymbol{K}=\{(n,p,q,a,b)\mid n=pq,p,q$ 为素数，$ab\equiv1(\mathrm{mod}\ \varphi(n))\}$。值 n 和 b 是公开的，值 p,q 和 a 是保密的。

对 $K=(n,p,q,a,b)$，定义 $\mathrm{Sig}_K(x)=x^a \bmod n, x\in Z_n$

$$\mathrm{Ver}_K(x,y)=\text{真}\Leftrightarrow x\equiv y^b(\mathrm{mod}\ n)\qquad x,y\in Z_n.$$

如果 B 使用 RSA 解密规则 D_K 签一个消息 x，那么 B 是能产生签名的唯一的人，这是因为 D_K 是保密的。验证算法使用 RSA 加密规则 E_K。因为 E_K 是公开的，所以任何人能

验证一个签名。

RSA 数字签名方案的弱点如下：(1)任何人能通过对某一 y 计算 $x=E_K(y)$ 伪造一个随机消息 x 关于 B 的签名 y，这是因为 $y=\mathrm{Sig}_K(x)$；(2)如果消息 x_1 和 x_2 的签名分别是 y_1 和 y_2，则拥有 x_1, y_1, x_2, y_2 的任何人可伪造 B 关于消息 x_1x_2 的签名 y_1y_2，这是因为 $\mathrm{Sig}_K(x_1x_2)=\mathrm{Sig}_K(x_1)\mathrm{Sig}_K(x_2)\bmod n$；(3)签名者 B 每次只能签$[\log_2 n]$比特长的消息，获得同样长的签名。一般来说，所要签的消息都很长，如果直接对消息进行签名只能先把所要签的消息分成$[\log_2 n]$比特长的组，逐组进行签名。因为 RSA 数字签名中所涉及的运算都是模乘法运算和模指数运算，所以这样做带来的缺陷是速度慢，而且签名太长。

克服上述三个缺陷的办法之一是在对消息进行签名之前作变换，这种变换并非是一般的变换，而是杂凑(hash)变换，亦称杂凑函数；关于杂凑函数的研究参见第 8 章。

7.1.2 加密和签名的结合

本小节我们来说明，当加密和签名相结合时，加密和签名过程的顺序不一样将导致不同的安全程度。这就告诫我们结合使用加密和签名技术时要特别小心。

如果要求 A 给 B 发送消息时不仅要对消息进行签名而且还要对消息进行加密，那么在这种情况下，A 可采用如下两种方式发送消息：一种是先签名，后加密；另一种是先加密，后签名。

给定明文消息 x，在第一种方式中，A 先对 x 进行签名，$y=\mathrm{Sig}_A(x)$，然后使用 B 的公开加密变换对 x 和 y 进行加密，$z=E_B(x,y)$，最后 A 将密文 z 传给 B。当 B 收到 z 时，他首先利用他的解密变换解密 z 获得(x,y)，然后他使用 A 的公开验证函数检测是否有 $\mathrm{Ver}_A(x,y)=$真。在第二种方式中，A 先利用 B 的公开加密变换对 x 进行加密，$z=E_B(x)$，然后对 z 进行签名，$y=\mathrm{Sig}_A(z)$，最后 A 将(z,y)发送给 B。B 将解密 z 获得 x，然后使用 Ver_A验证 A 对 x 的签名 y。用第二种方式存在一个问题，如果一个敌手 O 获得一对(z,y)，他可用 $y'=\mathrm{Sig}_O(z)$代替 y 得到一对(z,y')，将(z,y')发送给 B，B 先对 z 解密，然后使用 O 的验证算法 Ver_O验证签名的合法性。这样 B 误认为消息 x 来自 O。由于这种潜在的危险，大多数人建议采用第一种方式，即先签名后加密。

7.2 ElGamal 型数字签名方案和数字签名标准(DSS)

ElGamal 于 1985 年基于离散对数问题提出了一个既可用于加密又可用于数字签名的密码体制。这个数字签名方案的一个修改已被美国国家标准技术研究所(NIST)采纳为数字签名标准。ElGamal 数字签名方案像 ElGamal 公钥密码算法一样是非确定性的。这意味着对任何给定的消息有许多合法的签名。ElGamal 密码体制是基于 Z_p^*(p 为素数)上的离散对数问题，可以将 ElGamal 密码体制一般化到基于任意有限群上的离散对数问题[7.8]。本节我们只考虑基于 Z_p^* 上的离散对数问题的 ElGamal 数字签名方案。首先我们来描述一个一般的数字签名方案。

设 p 是一个使得在 Z_p 上的离散对数问题是难处理的素数，q 是 $p-1$ 的一个大素数因子或 $q=p$，当 $q<p$ 时，随机选择一个阶为 q 的元素 $\alpha\in Z_p^*$；当 $q=p$ 时，随机选择一个阶为 $p-1$ 的元素 $\alpha\in Z_p^*$，$\boldsymbol{P}=Z_q^*$，$\boldsymbol{A}=Z_q^*\times Z_q$，$(q<p)$或 $\boldsymbol{A}=Z_p^*\times Z_{p-1}(q=p)$定义

$$K = \{(p,q,\alpha,a,\beta) \mid \beta \equiv \alpha^a (\bmod\ p)\}$$

值 p,q,α 和 β 是公开的，a 是保密的。

对 $K=(p,q,\alpha,a,\beta)$ 和一个秘密的随机数 $k\in Z_q^*$ $(q<p)$ 或 $k\in Z_{p-1}^*$ $(q=p)$ 定义 $\mathrm{Sig}_K(x,k)=(\gamma,\delta)$。其中，$\gamma=\alpha^k \bmod p \pmod q$。

当 $q<p$ 时，δ 满足方程

$$k\cdot f(\gamma,x,\delta)+a\cdot g(\gamma,x,\delta)+h(\gamma,x,\delta)\equiv 0(\bmod\ q) \tag{7.2.1}$$

当 $q=p$ 时，δ 满足方程

$$k\cdot f(\gamma,x,\delta)+a\cdot g(\gamma,x,\delta)+h(\gamma,x,\delta)\equiv 0(\bmod\ p-1) \tag{7.2.2}$$

f,g,h 是公开知道的函数，并且从式(7.2.1)或(7.2.2)求 δ 是容易的。

对 $x\in Z_p^*$，$\gamma\in Z_q^*$ 和 $\delta\in Z_q$，定义

$$\mathrm{Ver}(x,\gamma,\delta)=\text{真}\Leftrightarrow \gamma^{f(\gamma,x,\delta)}\cdot\beta^{g(\gamma,x,\delta)}\cdot\alpha^{h(\gamma,x,\delta)}\equiv 1(\bmod\ p)(\bmod\ q) \tag{7.2.3}$$

因为 f,g,h 是公开知道的函数，并且 (x,γ,δ) 是公开的，所以任何人都可验证式(7.2.3)。

如果签名被正确地构造，那么验证将是成功的，因为 $\gamma^{f(\gamma,x,\delta)}\cdot\beta^{g(\gamma,x,\delta)}\cdot\alpha^{h(\gamma,x,\delta)}(\bmod\ p)(\bmod\ q)=\alpha^{kf(\gamma,x,\delta)}\cdot\alpha^{ag(\gamma,x,\delta)}\cdot\alpha^{h(\gamma,x,\delta)}(\bmod\ p)(\bmod\ q)=1$

当 f,g,h 取不同的函数时，就可得出不同的数字签名方案，我们统称这一类方案为 ElGamal 型数字签名方案。文献[3,9,10,11]的数字签名方案均为这种类型。下面我们来介绍两个典型的例子。

7.2.1 ElGamal 数字签名方案

在上述方案中，当取 $q=p$，$f(\gamma,x,\delta)=\delta$，$g(\gamma,x,\delta)=\gamma$，$h(\gamma,x,\delta)=-m$ 时就是 ElGamal 数字签名方案[3]。

此时，签名算法和验证算法分别为：$\mathrm{Sig}_K(x,k)=(\gamma,\delta)$，$\gamma=\alpha^k \bmod p$，$\delta=(x-a\gamma)k^{-1} \bmod p-1$；$\mathrm{Ver}(x,\gamma,\delta)=$真$\Leftrightarrow\beta^\gamma\gamma^\delta\equiv\alpha^x(\bmod\ p)$。$x,\gamma\in Z_p^*$，$\delta\in Z_{p-1}$。

现在我们来看一看 ElGamal 数字签名方案的安全性。假定 O 不知道 a，他现在想伪造一个给定的消息 x 的数字签名。如果 O 首先选择一个值 γ，然后极力找一对应的 δ，那么他必须计算离散对数 $\log_\gamma(\alpha^x\beta^{-\gamma})$。如果 O 首先选择 δ，然后极力找到 γ，那么他必须通过解方程 $\beta^\gamma\gamma^\delta\equiv\alpha^x(\bmod\ p)$ 来求出 γ。目前还没有可行的办法来解这个方程，这个问题似乎与离散对数问题没什么关系。也许有一种方法可同时计算 γ 和 δ，使得 (γ,δ) 是 x 的签名，然而，至今还没有人发现解这个问题的方法，也没有人能证明不能解这个问题。

如果 O 首先选择 γ 和 δ，然后极力去解 x，那么他不得不计算离散对数 $\log_\alpha\beta^\gamma\gamma^\delta$。因此，O 不能使用这种方法签一个随机消息。然而对 ElGamal 数字签名方案有以下两种类型的伪造攻击。

伪造类型 1：O 可通过同时选择 γ,δ 和 x 来签一个随机消息。

设 i,j 是整数，$0\leqslant i\leqslant p-2$，$0\leqslant j\leqslant p-2$，$\gcd(j,p-1)=1$，计算 $\gamma=\alpha^i\beta^j \bmod p$，$\delta=-\gamma j^{-1} \bmod (p-1)$，$x=-\gamma ij^{-1}(\bmod\ (p-1))$，这里 j^{-1} 是 j 关于模 $p-1$ 的逆。直接可验证 (γ,δ) 是 x 的一个合法签名。

伪造类型 2：假定 (γ,δ) 是 B 对消息 x 的一个合法签名，O 从这个签名消息开始可签各种别的消息。设 h,i 和 j 是整数，$0\leqslant h,i,j\leqslant p-2$，$\gcd(h\gamma-j\delta,p-1)=1$，计算 $\lambda=\gamma^h\alpha^i\beta^j$

mod p，$\mu=\delta\lambda(h\gamma-j\delta)^{-1}\bmod(p-1)$，$x'=\lambda(hx+i\delta)\cdot(h\gamma-j\delta)^{-1}\bmod(p-1)$，这里$(h\gamma-j\delta)^{-1}$是$(h\gamma-j\delta)$关于模$p-1$的逆。直接可验证$(\lambda,\mu)$是$x'$的一个合法签名。

虽然上述两种方法都可产生许多伪造的签名，但是似乎一个敌手除了求离散对数之外还没有办法伪造一个他自己选择的消息的一个合法签名。因此，这些方法似乎对 ElGamal 数字签名方案的安全性并未构成威胁。另外，对消息进行签名前先对消息作杂凑变换可克服这些缺陷。

在使用 ElGamal 型数字签名方案时，应特别注意两个问题，一个是不能将签名中所使用的随机值 k 泄露出去；另一个是不能使用同一个 k 值签两个不同的消息。现以 ElGamal 数字签名方案来说明这样做所带来的隐患。

如果将签名中所使用的随机值 k 泄露，那么知道 k 的任何人可由方程 $\delta=(x-a\gamma)k^{-1}\bmod(p-1)$求出 $a=(x-\delta k)\gamma^{-1}\bmod(p-1)$。当然，一旦 a 被知道，那么系统就被破译，O 能随意地伪造签名。

如果系统使用同一个 k 值签了两个不同的消息，那么 O 可容易地计算出 a，从而破译该系统。计算 a 的过程如下：

假定(γ,δ_1)和(γ,δ_2)分别是 x_1 和 $x_2(x_1\neq x_2)$的合法签名(因为使用了同一个值 k，所以第一个分量相等，即 $\gamma_1=\gamma_2=\gamma$)，那么我们有 $\beta^{\gamma}\gamma^{\delta_1}\equiv\alpha^{x_1}(\bmod p)$，$\beta^{\gamma}\gamma^{\delta_2}\equiv\alpha^{x_2}(\bmod p)$。这样，$\alpha^{x_1-x_2}\equiv\gamma^{\delta_2-\delta_1}(\bmod p)$。令 $\gamma=\alpha^k$，则 $\alpha^{x_1-x_2}\equiv\alpha^{k(\delta_2-\delta_1)}(\bmod p)$，而这个方程等价于 $x_1-x_2\equiv k(\delta_2-\delta_1)(\bmod(p-1))$。

设 $d=\gcd(\delta_2-\delta_1,p-1)$，因为 $d|p-1$，$d|\delta_2-\delta_1$，所以由上式可知，$d|x_1-x_2$。令 $x'=\frac{x_1-x_2}{d}$，$\delta'=\frac{\delta_2-\delta x_1}{d}$，$p'=\frac{p-1}{d}$。此时上式同余式变为 $x'\equiv k\delta'(\bmod p')$。因为 $\gcd(\delta',p')=1$，所以 $k=x'\varepsilon\bmod p'$，$\varepsilon=(\delta')^{-1}\bmod p'$。这就产生了 d 个候选的 k 值，$k=(x'\varepsilon+ip')\bmod p$，$0\leqslant i\leqslant d-1$。可通过等式 $\gamma\equiv\alpha^k(\bmod p)$检测出唯一正确的 k 值。知道 k 值后便可求出 a，从而破译了整个系统。

文献[3]提出了一种新的伪造 ElGamal 数字签名的方法，它指出如果 ElGamal 数字签名方案的参数选择的不恰当的话，无需知道秘密密钥就可产生 ElGamal 数字签名。这并不是说我们能破译 ElGamal 数字签名方案，而是提醒我们在选择方案的参数时要特别小心。

7.2.2 数字签名标准(DSS)

当 $q<p$，$f(\gamma,x,\delta)=\delta$，$g(\gamma,x,\delta)=-\gamma$，$h(\gamma,x,\delta)=-H(x)$时，ElGamal 型数字签名方案就是数字签名标准(DSS)[11]，其中 H 是一个杂凑(hash)函数。

此时，签名算法和验证算法分别为

$$\mathrm{Sig}_K(x,k)=(\gamma,\delta)\qquad \gamma=(\alpha^k\bmod p)\bmod q\qquad \delta=(H(x)+a\gamma)k^{-1}\bmod q$$

和

$$\mathrm{Ver}(x,\gamma,\delta)=\text{真}\Leftrightarrow(\alpha^{e_1}\beta^{e_2}\bmod p)\bmod q=\gamma$$

$$e_1=x\delta^{-1}\bmod q\qquad e_2=\gamma\delta^{-1}\bmod q\quad x\in Z_p^*\qquad \gamma,\delta\in Z_q$$

DSS 是由美国国家标准技术研究所(NIST)于 1991 年 8 月 30 日提出，1994 年 5 月 19 日在联邦记录(FR)中公布，1994 年 12 月 1 日被采纳的一个数字签名标准。它是 ElGamal 数字签名方案的一个变形。该标准最初版中建议使用 p 为 512 比特的素数，q 为 160

比特的素数，$q|(p-1)$。后来在众多的批评下，NIST 建议使用长为 512 比特到 1024 比特且其长度是 64 的倍数的素数 p，当 p 选为 512 比特的素数时，ElGamal 签名的长度为 1024 比特，而在 DSS 中通过一个 160 比特的素数可将签名的长度降低为 320 比特，这就大大地减少了存贮空间和传输带宽。不能不说这是 DSS 的一个天才的改进。

在 DSS 的验证算法中，我们使用了 $\delta^{-1} \bmod q$，这就要求 $\delta \neq 0 (\bmod q)$。这一点可通过下述办法解决：在 DSS 的签名算法中，如果 $\delta \equiv 0 (\bmod q)$，那么签名者重新选择一个 k 构造一个新的签名。这样做不会产生什么问题，这是因为在实际应用中，$\delta \equiv 0 (\bmod q)$ 的概率很小，大约为 2^{-160}，所以可认为这种情况从来不发生。

现在我们来谈谈 DSS 的实现。在 DSS 方案中涉及到的求逆运算可用 Euclidean 算法来完成。求模指数运算可用“平方-乘”算法来完成。

阶为 q 的元素 $\alpha \in Z_p^*$ 可根据下列步骤来选取：(1)随机选择一个元素 $g \in Z_p^*$ 计算 $\alpha = g^{(p-1)/q} \bmod p$；(2)如果 $\alpha \neq 1$，那么 α 就是所要选取的阶为 q 的元素，否则重新回到步骤(1)。实际实现 DSS 时可通过预计算来加快速度。因为 γ 的值与消息无关，所以可以事先产生一串随机的 k 并且预先计算出与之对应的 γ 值，还可以对每个 k 值预先计算出 k^{-1}。这样，一旦签名消息 x 时，只要由 x 及已给定的 γ 和 k^{-1} 计算出 δ 即可。现在我们关心的主要问题是如何来选择 DSS 中的素数 p 和 q，$q|(p-1)$，p 是长度为 512 比特到 1024 比特且其长度是 64 的倍数的素数，q 是长度为 160 比特的素数。NIST 推荐了一种产生 DSS 中的素数 p 和 q 的方法，其详细产生过程如下：

设 $L=512+64l$，$0 \leqslant l \leqslant 8$，$H(\cdot)$ 是一个 Hash 函数，这里的 H 可选为 SHA-1(详细描述参见第 8 章)。

输入：一个整数 l，$0 \leqslant l \leqslant 8$。

(1) 计算 $L=512+64l$，使用 Euclidean 除法(长除法)找到 n 和 b 使得 $L-1=160n+b$，$0 \leqslant b < 160$。

(2) 重复下列步骤：

(a)选择一个至少为 160 比特长的随机序列 S，记 S 的长度为 g，$g \geqslant 160$。

(b)计算 $\mu = H(S) \oplus H((S+1) \bmod 2^g)$，在 $H((S+1) \bmod 2^g)$ 的计算中，先将 S 视作一个整数，其次与 1 模 2^g 相加，最后，再将所加的结果视作一个二元表示。究竟将 x 视作一个整数 $x=x_{g-1}2^{g-1}+\cdots+x_1 2+x_0$ $(0 \leqslant x < 2^g)$，还是视作一个 g 比特串 $x_{g-1}x_{g-2}\cdots x_1 x_0$，视具体情况而定。

(c)通过对 μ 的最高位和最低位置 1 获得一个 160 比特的奇整数 q。

(d)检测 q 是否为素数。

(e)如果不是素数，重新回到步骤(a)，直到找到一个 160 比特的素数 q 为止。

(3)置 $i=0$，$j=2$。

(4) 当 $i<4096$ 时，完成下列步骤：

(a)对 $k=0$ 到 n，置 $V_k = H((s+j+k) \bmod 2^g)$。

(b)置 $W=V_0+V_1 2^{160}+V_2 2^{320}+\cdots+V_{n-1}2^{160(n-1)}+(V_n \bmod 2^b)2^{160n}$，$X=W+2^{L-1}$ (X 是一个 L 比特长的整数)。

(c)计算 $c=X \bmod 2q$，置 $p=X-(c-1)$ (注：$p \equiv 1 (\bmod 2q)$)。

(d)如果 $p<2^{L-1}$，转到步骤(e)，否则，检测 p 是否为素数，如果 p 为素数，输出 (p,q)

并存储(S,i)。

(e)置$i=i+1,j=j+n+1$。

(5)返回步骤(2)。

输出:一个160比特长的素数q和一个L比特长的素数p,并且$q|(p-1),L=512+64l$。

这表明存在着公开的方法来产生p和q。如果某人给你一个p和一个q,你可能会想知道他们是从哪里得到它们的。但是,如果某人给你产生随机的p和q的S和i的值,你就可以自己运行上述算法。单向函数$H(\cdot)$的使用能阻止他人在背后做手脚。这样做的安全性比RSA体制的安全性要高。在RSA体制中,素数是秘密保存的。某人可能产生假素数或容易分解的特殊形式的素数,除非你知道秘密密钥,否则你不知道这一点。而在这里,即使你不知道秘密密钥,你也可以确信p和q是随机产生的。

接下来我们谈谈DSS出台后人们对它的批评意见。DSS在评论阶段收到来自政府、工业和个人等方面的100多篇评论[12]。关于DSS的一些批评意见主要表现在以下几个方面:

(1)许多人抱怨NIST选择数字签名标准的过程不是公开的,而且标准没有美国工业界的介入,而是由美国国家安全局(NSA)提出的。不管产生的方案有多少优点,人们讨厌这种"关门"做法。

(2)许多人认为把模p的长度固定为512比特是不合适的,从长远的角度来看是不安全的,希望模p的长度是可变的,根据需要而定。在评论的答复中,NIST改变了标准的描述,允许模的长度是可变的。

(3)在DSS中,签名比验证快得多。相反,在用RSA签名时,如果选用较小的公开验证指数,那么验证比签名快得多。NIST认为在签名和验证都充分快的情况下,签名比验证快并不会影响任何事情。

(4)q的长度为160比特有点太短。

(5)在DSS中需求$\delta^{-1}\bmod q$,因而要求$\delta \bmod q \neq 0$。NIST认为在实际应用中这不会产生问题,因为可约定$\delta \bmod q=0$时,接收者拒绝接收签名,要求利用新的随机数重新构造签名,而且$\delta \bmod q=0$的概率相当小。

(6)与RSA算法相比,DSS不是一个双射函数,并且不能用于加密和密钥分配。许多DSS的批评者认为,一个基于RSA的数字签名工具的广泛获得必将导致一个安全的加密机制,而NSA为了避免这种情况才选择了一个不可逆函数。但NIST的答复中指出,DSS的不可逆性恰是一个安全特征,并说明RSA的可逆性在一些协议中也能提供攻击的途径。

关于DSS的安全性的讨论可参阅文献[14,15,16]。另外,因为DSS是ElGamal签名方案的一个变形,所以有关ElGamal签名方案的一些伪造攻击也可能对DSS奏效。不过已有的这些关于ElGamal型数字签名方案的攻击方法并没有对其构成威胁。提醒读者值得注意的是在使用密码方案之前应先查阅一下有关的最新材料,看一下选用的方案是否已被破译或有哪些漏洞以便改进和弥补某些不足之处。

基于DSS设计的一些协议和签名方案请参阅文献[9,17,18]。

7.3 一次数字签名方案

本节我们来介绍从任何单向函数构造一次数字签名方案的一种方法。术语"一次"意味着只能签一个消息，当然，对签名可进行任意次验证。现在我们来描述从任何单向函数构造一次数字签名方案的一个具体过程，所构作的方案称作 Lamport 数字签名方案[5]。

设 k 是一个正整数，$P=\{0,1\}^k$。假定 $f:Y\to D$ 是一个单向函数，$A=Y^k$。在 Y 中随机选择 $2k$ 个元素 $y_{i,j}, 1\leqslant i\leqslant k, j=0,1$，设 $z_{i,j}=f(y_{i,j}), 1\leqslant i\leqslant k, j=0,1$。密钥 K 由 $2k$ 个 $y_{i,j}$ 和 $2k$ 个 $z_{i,j}$ 构成。$\{y_{i,j}\}_{\substack{1\leqslant i\leqslant k\\ j=0,1}}$ 是保密的，$\{z_{i,j}\}_{\substack{1\leqslant i\leqslant k\\ j=0,1}}$ 是公开的。

对 $K=\{y_{i,j},z_{i,j}\mid 1\leqslant i\leqslant k, j=0,1\}$，定义

$$\mathrm{Sig}_K(x_1,x_2,\cdots,x_k)=(y_{1x_1},\cdots,y_{k,x_k})=(a_1,a_2,\cdots,a_k),(x_1,x_2,\cdots,x_k)\in P$$

$$\mathrm{Ver}_K(x_1,x_2,\cdots,x_k,a_1,a_2,\cdots,a_k)=\text{真}\Leftrightarrow f(a_i)=z_{i,x_i}\qquad 1\leqslant i\leqslant k$$

$$(a_1,a_2,\cdots,a_k)\in A$$

方案中的单向函数 f 可由不同的途径来构造。可基于某些已知的困难问题，比如离散对数问题来构造。设 p 是一个大素数，α 是 Z_p 的一个本原元，$f(x)=\alpha^x \bmod p$ 是一个单向函数。也可基于私钥密码体制来构造，因为一般来讲，一个私钥密码体制在已知明文或选择明文攻击下要推出密钥是相当难的。比如，令明文为 m，密文为 c，密钥为 k，加密变换为 E_k，即 $c=E_k(m)$。由于从 c 和 m 难于求得密钥 k，所以选择一个常数 a 用密钥 x 加密不难得到 $y=E_x(a)$，由此定义一单向函数 $f(x)=y$。在已知 y 时要推出 x 就等价于已知明文 a 和 y 的条件下确定 x。这也表明，利用私钥密码体制进行数字签名是可行的。

敌手 O 不能伪造一个签名，因为他不能通过求单向函数 f 的逆来获得秘密 $y_{i,j}$。然而，该签名方案只能签一个消息。给定两个不同消息的签名，敌手很容易构造出不同于这两个消息的别的消息的签名。例如，假定消息(0,1,1)和(1,0,1)都使用同一个签名方案签名，消息(0,1,1)的签名为$(y_{1,0},y_{2,1},y_{3,1})$，消息(1,0,1)的签名为$(y_{1,1},y_{2,0},y_{3,1})$。给定这两个签名后，敌手 O 就能够构造消息(1,1,1)和(0,0,1)的签名，(1,1,1)的签名为$(y_{1,1},y_{2,1},y_{3,1})$，(0,0,1)的签名为$(y_{1,0},y_{2,0},y_{3,1})$。

从 Lamport 方案产生的签名的长度来看，该方案不太实用。例如，如果利用单向函数 $f(x)=\alpha^x \bmod p$，则根据安全性要求，p 至少是 512 比特长，这意味着消息的每一个比特的签名是 512 比特。因此，签名的长度是消息的长度的 512 倍。

在 Lamport 方案中，给定消息的一个签名，敌手 O 不能伪造另一个消息的签名的原因是一个消息所对应的 $S_1=\{y_{i,j}\}$ 从来不是另一个消息所对应的 $S_2=\{y_{i,j}\}$ 的子集。Bos 和 Chaum 注意到了这一点，在不失安全性的条件下，他们对 Lamport 方案作了修改，使得签名较短。

假定 $\boldsymbol{B}$ 是集合 B 的某些子集所作成的集合。如果对所有的 $B_1,B_2,\in\boldsymbol{B}B_1\subseteq B_2$，必有 $B_1=B_2$，则我们称集合 $\mathbf{B}$ 满足 Sperner 特性。给定一个集合 B，$|B|=2n$，则满足 Sperner 特性的最大集合 $\mathbf{B}$($\mathbf{B}$ 由 B 的子集构成)的个数为 C_{2n}^n。这一最大集合 $\mathbf{B}$ 可由下列办法容易构造出：取 B 的所有 n 个元素的子集(称为 n-子集)所构成的集合作为 $\mathbf{B}$。显然，$\mathbf{B}$ 中的任何一个 n-子集都不包含在另一个 n-子集之中，且 $|\mathbf{B}|=C_{2n}^n$。

现在我们来描述 Bos-Chaum 数字签名方案[19]。

设 k 是一个正整数，$P=\{0,1\}^k$，n 是满足条件 $2^k\leqslant C_{2n}^n$ 的一个正整数。B 是一个含 $2n$ 个元素的集合。设 $\phi:\{0,1\}^k\to\mathbf{B}$ 是一个单射，$\mathbf{B}$ 是由 B 的所有 n-子集所作成的集合。假定 $f:Y\to D$ 是一个单向函数。$A=Y^n$。随机地从 Y 中选择 $2n$ 个元素 y_i，$1\leqslant i\leqslant 2n$，并设 $z_i=f(y_i)$，$1\leqslant i\leqslant 2n$。密钥 K 由 $\{y_i\}_{i=1}^{2n}$ 和 $\{z_i\}_{i=1}^{2n}$ 组成。$\{y_i\}_{i=1}^{2n}$ 是保密的，而 $\{z_i\}_{i=1}^{2n}$ 是公开的。f 和 ϕ 都是公开知道的。

对 $K=\{y_i,z_i\mid 1\leqslant i\leqslant 2n\}$，定义

$$\mathrm{Sig}_K(x_1,x_2,\cdots,x_k)=\{y_j\mid j\in\phi(x_1,x_2,\cdots,x_k)\}\qquad(x_1,x_2,\cdots,x_k)\in P$$

$$\mathrm{Ver}_K(x_1,x_2,\cdots,x_k,a_1,a_2,\cdots,a_n)=\text{真}$$

$$\Leftrightarrow\{f(a_i)\mid 1\leqslant i\leqslant n\}=\{z_j\mid j\in\phi(x_1,\cdots,x_n)\}\qquad(a_1,a_2,\cdots,a_n)\in A$$

Bos-Chaum 方案与 Lamport 方案相比，它的优点是签名较短。例如我们想签一个 6 比特的消息，即 $k=6$。因为 $2^6=64$，$C_8^4=70$，所以我们取 $n=4$。这表明，在 Bos-Chaum 方案中，一个 6 比特的消息可用 4 个 y_i 来签名。而在 Lamport 方案中，一个 6 比特的消息需用 6 个 y_i 来签名。另外 Bos-Chaum 方案中的密钥也比 Lamport 方案中的密钥短。

Bos-Chaum 方案中需要一个单射 ϕ 将一个 $2n$-集合的 n-子集和每一个可能的二元 k 维向量 $x=(x_1,x_2,\cdots,x_k)$ 联系起来。现在我们给出计算 ϕ 的一个简单算法：

(1) $x=\sum_{i=1}^{k}x_i2^{i-1}$；

(2) $\phi(x)=\varnothing$；

(3) $t=2n$；

(4) $e=n$；

(5) while $t>0$ do

(6) $t=t-1$；

(7) If $x>\binom{t}{e}$ then

(8) $x=x-\binom{t}{e}$；

(9) $e=e-1$；

(10) $\phi(x)=\phi(x)\cup\{t+1\}$。

在 Bos-Chaum 方案中，要求 $2^k\leqslant C_{2n}^n$。因为 $C_{2n}^n=\frac{(2n)!}{(n!)^2}$，由 Stirling 公式，我们有 $C_{2n}^n\approx 2^{2n}/\sqrt{\pi n}$。这样 $2^k\leqslant 2^{2n}/\sqrt{\pi n}$，即 $k\leqslant 2n-\frac{\log_2(n\pi)}{2}$。渐近地来看，$n$ 大约为 $k/2$，所以使用 Bos-Chaum 方案签名使得签名的长度大约降低 50%。

7.4 不可否认的数字签名方案

不可否认的数字签名[20](undeniable digital signature)是由 Chaum 和 Antwerpen 在 1989 年提出的。不可否认的数字签名是一种特殊的数字签名，它具有一些新颖的特征，没有签名者的合作，接收者就无法验证签名，在某种程度上这种签名保护了签名者的利益。

这种签名在某些应用场合是十分有用的，诸如软件开发者可利用不可否认的数字签名对他们的软件进行保护，使得只有付了钱的顾客才能验证签名并相信开发者仍然对软件负责。

一个不可否认的签名的真伪性是通过接收者和签名者执行一个协议来推断的（假定签名者参加了这个协议），这个协议称作否认协议(disavowal protocol)。因为签名者可声称一个合法的签名是伪造的，在这种情况下，如果签名者拒绝参加验证，我们就可认为签名者有欺骗行为。如果签名者参加验证，由否认协议就可推断出签名者的真伪性。自从不可否认的数字签名诞生以来，人们围绕这种签名讨论了一系列相关的签名，诸如可转移的不可否认的数字签名、零知识的不可否认的数字签名等，感兴趣的读者请参阅文献[21,22,23,24]。

一个不可否认的数字签名方案由以下三部分组成：一个签名算法，一个验证协议和一个否认协议。现在我们先来介绍 Chaum-van Antwerpen 不可否认的数字签名方案[20]的签名算法和验证协议。

设 $p=2q+1$ 是一个使得 q 是素数并且在 Z_p 上的离散对数是难处理的素数。$\alpha\in Z_p^*$ 是一个阶为 q 的元素，$1\leqslant a\leqslant q-1$，令 $\beta=\alpha^a \bmod p$。设 G 表示 Z_p^* 中的阶为 q 的乘法子群（G 由模 p 的二次剩余构成）。设 $\boldsymbol{P}=\boldsymbol{A}=G$，定义 $\boldsymbol{K}=\{(p,\alpha,a,\beta)\mid\beta\equiv\alpha^a(\bmod p)\}$。值 p，α 和 β 是公开的，a 是保密的。

对 $K=(p,\alpha,a,\beta)$和 $x\in G$，定义 $y=\mathrm{Sig}_K(x)=x^a \bmod p$。对 $x,y\in G$，通过执行下列协议来验证签名：

(1) 接收者 A 随机选择 $e_1,e_2\in Z_q^*$；

(2) A 计算 $c=y^{e_1}\beta^{e_2}\bmod p$，并将 c 发送给签名者 B；

(3) B 计算 $d=c^{a^{-1}\bmod q}\bmod p$，并将 d 发送给 A；

(4) A 将 y 作为合法的签名接收$\Leftrightarrow d\equiv x^{e_1}\alpha^{e_2}(\bmod p)$。

下面我们来证明 B 只能以很小的概率欺骗 A。这个结果不依赖于任何计算假设，也就是说，安全性是无条件的。

定理 7.4.1 如果 $y\neq x^a(\bmod p)$，那么接收者 A 将以 $1/q$ 的概率把 y 作为 x 的一个合法签名接收。

证明：因为 y 和 β 都是阶为素数 q 的乘法群 G 中的元素，所以每一个可能的口令 c 恰好对应于 q 个有序对(e_1,e_2)。当签名者 B 接收到口令 c 时，他没有办法知道 c 是 A 用这 q 个可能的有序对(e_1,e_2)中的哪一个来构造的。下面我们来说明，如果 $y\neq x^a(\bmod p)$，那么 B 所作的任何可能的回答 $d\in G$ 恰好和这 q 个有序对(e_1,e_2)中的一个一致。

因为 α 是 G 的生成元，所以我们能将 G 中的每一个元素表示成 α 的幂次方。设 $c=\alpha^j$，$d=\alpha^l$，$x=\alpha^k$，$y=\alpha^i$，$i,j,k,l\in Z_q$。考虑下列两个同余式：

$$c\equiv y^{e_1}\beta^{e_2}(\bmod p)\qquad d\equiv x^{e_1}\alpha^{e_2}(\bmod p)$$

上面两个同余式等价于下面两个同余式：

$$\left.\begin{aligned} i&\equiv le_1+ae_2(\bmod q)\\ j&\equiv ke_1+e_2(\bmod q)\end{aligned}\right\}\tag{7.4.1}$$

现在我们假定 $y\neq x^a(\bmod p)$，这等价于 $l\neq ak(\bmod q)$。而同余方程组(7.4.1)的系数矩

阵为$\begin{pmatrix} l & a \\ k & 1 \end{pmatrix}$，该矩阵的行列式$\begin{vmatrix} l & a \\ k & 1 \end{vmatrix} = l - ak \not\equiv 0 \pmod{q}$，所以方程组(7.4.1)有唯一解。也就是说，每一个$d \in G$恰好是对q个可能的有序对(e_1, e_2)中之一的正确回答。因此，B将恰以$1/q$的概率给A一个能通过验证的回答d。

现在我们来介绍Chaum-van Antwerpen不可否认的签名方案的否认协议。该协议由两轮验证协议组成。步骤(1)～(4)和步骤(5)～(8)构成两轮不成功的验证协议。步骤(9)是一个一致性检测(consistency check)，这个检测能使A确定是否B按协议中规定的方式形成他的回答。否认协议的详细执行过程如下：

(1) A随机选择$e_1, e_2 \in Z_q^*$；

(2) A计算$c = y^{e_1}\beta^{e_2} \bmod p$并将$c$发送给B；

(3) B计算$d = c^{a^{-1} \bmod q} \bmod p$并将$d$发送给A；

(4) A验证$d \not\equiv x^{e_1}\alpha^{e_2} \pmod{p}$；

(5) A随机选择$f_1, f_2 \in Z_q^*$；

(6) A计算$C = y^{f_1}\beta^{f_2} \bmod p$并将$C$发送给B；

(7) B计算$D = C^{a^{-1} \bmod q} \bmod p$并将$D$发送给A；

(8) A验证$D \not\equiv x^{f_1}\alpha^{f_2} \pmod{p}$；

(9) A作出推断：y是一个伪造$\Leftrightarrow (d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \pmod{p}$。

我们现在需说明否认协议可做到以下两点要求：一是B能使A相信一个不合法的签名是一个伪造；二是B以一个很小的概率使A相信一个合法的签名是一个伪造。

定理7.4.2 如果$y \not\equiv x^a \pmod{p}$，并且A和B都采纳了否认协议，那么$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \pmod{p}$。

证明：因为$d \equiv c^{a^{-1}} \pmod{p}$，$c \equiv y^{e_1}\beta^{e_2} \pmod{p}$，$\beta \equiv \alpha^a \pmod{p}$，所以我们有

$$\begin{aligned}(d\alpha^{-e_2})^{f_1} &\equiv ((y^{e_1}\beta^{e_2})^{a^{-1}}\alpha^{-e_2})^{f_1} \pmod{p} \\ &\equiv y^{e_1 f_1 a^{-1}}\beta^{e_2 a^{-1} f_1}\alpha^{-e_2 f_1} \pmod{p} \\ &\equiv y^{e_1 f_1 a^{-1}}\alpha^{e_2 f_1}\alpha^{-e_2 f_1} \pmod{p} \\ &\equiv x^{e_1 f_1} \pmod{p}\end{aligned}$$

同理可计算出，$(D\alpha^{-f_2})^{e_1} \equiv x^{e_1 f_1} \pmod{p}$。所以否认协议中步骤(9)的一致性检验是成功的。

定理7.4.2说明了否认协议可做到上述第一点要求。

B也许企图否认一个合法的签名。在这种情况下，我们不假定B采纳了协议，也就是说，B也许没有按协议的规定去构造d和D。因此，在下面的定理7.4.3中我们只假定B能产生满足否认协议中步骤(4)、(8)和(9)的值d和D。

定理7.4.3 假定$y \equiv x^a \pmod{p}$，并且A采纳了否认协议。如果$d \not\equiv x^{e_1}\alpha^{e_2} \pmod{p}$并且$D \not\equiv x^{f_1}\alpha^{f_2} \pmod{p}$，那么$(d\alpha^{-e_2})^{f_1} \not\equiv (D\alpha^{-f_2})^{e_1} \pmod{p}$的概率是$1 - 1/q$。

证明：假定下列的同余式成立：

$$\begin{aligned} y &\equiv x^a \pmod{p} \\ d &\not\equiv x^{e_1}\alpha^{e_2} \pmod{p} \\ D &\not\equiv x^{f_1}\alpha^{f_2} \pmod{p} \\ (d\alpha^{-e_2})^{f_1} &\equiv (D\alpha^{-f_2})^{e_1} \pmod{p}\end{aligned}$$

我们将导出一个矛盾。

一致性检测(步骤(9))可表示成下列形式：$D\equiv d_0^{f_1}\alpha^{f_2}(\bmod p)$，其中 $d_0=d^{1/e_1}\alpha^{-e_2/e_1}\bmod p$ 是仅仅依赖于否认协议中步骤(1)～(4)的一个值。

由定理 7.4.1 可知，y 是 d_0 的合法签名的概率为 $1-1/q$。但是我们已假定 y 是 x 的合法签名，所以 $x^a\equiv d_0^a(\bmod p)$的概率为 $1-1/q$，这暗含着 $x=d_0$ 的概率为 $1-1/q$。

然而，事实上 $d\equiv x^{e_1}\alpha^{e_2}(\bmod p)$意味着 $x\equiv d^{1/e_1}\alpha^{-e_2/e_1}(\bmod p)$。因为 $d_0\equiv d^{1/e_1}\alpha^{-e_2/e_1}(\bmod p)$，所以我们有 $x\neq d_0$，这是一个矛盾。

故 B 欺骗 A 的概率为 $1/q$。

定理 7.4.3 表明，否认协议可做到上述的第二点要求。

7.5 Fail-Stop 数字签名方案

Fail-Stop 数字签名[25,26]的不可伪造性也依赖于一个计算假设，但是如果一个签名被伪造，那么假定的签名者能证明这个签名是一个伪造签名。更精确地说，他能证明做基础的计算假设已经被攻破，这个证明伪造的能力不依赖于任何密码假设并独立于伪造者的计算能力，这样能保护一个多项式界的签名者免遭具有无限计算能力的伪造者的攻击。再者，在第一次伪造之后，系统的所有参加者或系统操作人员都知道签名方案已被攻破，因此系统将终止工作，这也是这个系统为什么称作"Fail-Stop"(失败-停止)的原因。

Fail-Stop 数字签名方案适用于联用的电子支付系统，在这个系统中可使顾客无条件地安全，顾客无需担心银行(一般来说银行比顾客有更强的计算能力)能攻破签名方案的基础假设。

一个 Fail-Stop 数字签名方案主要由三个算法，即签名算法、验证算法和"伪造证明"(proof of forgery)算法构成。一个安全的 Fail-Stop 数字签名方案应具有下列特性：

(1) 如果签名者正确地签一个消息，那么接收者接收这个签名；

(2) 一个多项式界的伪造者不能构造签名使之通过验证；

(3) 如果一个具有无限计算能力的伪造者成功地构造了一个签名使之通过验证，那么这个概率是极小的，签名者能产生一个伪造证明使得第三方相信一个伪造已经发生(这一特性是保证签名者的安全性)；

(4) 一个多项式界的签名者不能构造一个假签名使他后来证明是一个伪造(这一特性是保证接收者的安全性)。

达到这些特性的基本观点是相应于每个公钥有许多密钥(指数级的)并且对同一个消息不同的密钥给出不同的签名。签名者恰好知道这些密钥中的一个并且对一个给定的消息只能构造可能签名中的一个。然而，对一个新消息即使一个具有无限计算能力的伪造者也没有充分的信息确定签名者能构造众多可能的签名中的哪一个。因此，一个伪造签名以很高的概率不同于签名者已经构造的签名。而对同一个消息的两个不同签名的知识产生了一个伪造证明。

下面我们来介绍一个 Fail-Stop 数字签名方案，该方案是 Van Heyst 和 Pedersen 于 1992 年提出的一个一次签名方案[26]。首先我们描述 Van Heyst-Pedersen Fail-Stop 数字签名方案的签名算法和验证算法。

设 $p=2q+1$ 是一个使得 q 是素数并且在 Z_p 上的离散对数问题是难处理的素数，$\alpha\in Z_p^*$ 是一个阶为 q 的元素。设 $1\leqslant a_0\leqslant q-1$，定义 $\beta=\alpha^{a_0}\bmod p$。值 p,q,α,β 和 a_0 由一个可信中心来选取(这里假定系统中存在一个可信中心)。p,q,α 和 β 是公开的并且认为是固定的。a_0 的值对任何人(包括 B)都是保密的。

设 $P=Z_q,A=Z_q\times Z_q$。一个密钥 K 具有形式 $K=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$，这里 $a_1,a_2,b_1,b_2\in Z_q,\gamma_1=\alpha^{a_1}\beta^{a_2}\bmod p,\gamma_2=\alpha^{b_1}\beta^{b_2}\bmod p$。签名者 B 公开 γ_1,γ_2，保密 a_1,a_2,b_1,b_2。

对 $K=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$ 和 $x\in Z_q$，定义 $\mathrm{Sig}_K(x)=(y_1,y_2)$，这里 $y_1=(a_1+xb_1)\bmod q,y_2=(a_2+xb_2)\bmod q$。

对 $y=(y_1,y_2)\in Z_q\times Z_q$，我们有

$$\mathrm{Ver}_K(x,y)=\text{真}\Leftrightarrow\gamma_1\gamma_2^x\equiv\alpha^{y_1}\beta^{y_2}(\bmod p)$$

直接可验证，由签名者 B 产生的签名满足验证条件。下面我们将主要讨论该方案的安全性。首先，我们看一看有关密钥的一些事实。

我们说两个密钥 $K_1=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2)$ 和 $K_2=(\gamma_1',\gamma_2',a_1',a_2',b_1',b_2')$ 是等价的，意指 $\gamma_1=\gamma_1',\gamma_2=\gamma_2'$。易知，在任何一个等价类中恰有 q^2 个密钥。关于等价密钥我们有以下一些简单事实：

事实 1：假定 K 和 K' 是等价的密钥，并且 $\mathrm{Ver}_K(x,y)=$真，那么 $\mathrm{Ver}_{K'}(x,y)=$真。

证明：假定 $K=(\gamma_1,\gamma_2,a_1,a_2,b_1,b_2),K'=(\gamma_1',\gamma_2',a_1',a_2',b_1',b_2')$，这里 $\gamma_1'=\alpha^{a_1'}\beta^{a_2'}\bmod p=\alpha^{a_1}\beta^{a_2}\bmod p,\gamma_2'=\alpha^{b_1'}\beta^{b_2'}\bmod p=\alpha^{b_1}\beta^{b_2}\bmod p$。假定使用密钥 K 签消息 x 所得的签名为 (y_1,y_2)，这里 $y_1=(a_1+xb_1)\bmod q,y_2=(a_2+xb_2)\bmod q$。现在我们用密钥 K' 来验证 y：

$$\begin{aligned}\alpha^{y_1}\beta^{y_2}&\equiv\alpha^{a_1'+xb_1'}\beta^{a_2'+xb_2'}(\bmod p)\equiv(\alpha^{a_1'}\beta^{a_2'})(\alpha^{b_1'}\beta^{b_2'})^x(\bmod p)\\&\equiv\gamma_1'\gamma_2'^x(\bmod p)\end{aligned}$$

这表明，y 也可使用 K' 来验证，即 $\mathrm{Ver}_{K'}(x,y)=$真。

事实 2：假定 K 是一个密钥，并且 $y=\mathrm{Sig}_K(x)$，那么恰有 q 个与 K 等价的密钥 K' 使得 $y=\mathrm{Sig}_{K'}(x)$。

证明：假定 γ_1,γ_2 是 K 的公开部分。我们来考虑使得下列同余式组成立的四元组 (a_1,a_2,b_1,b_2) 的个数：

$$\begin{aligned}\gamma_1&\equiv\alpha^{a_1}\beta^{a_2}(\bmod p)\\\gamma_2&\equiv\alpha^{b_1}\beta^{b_2}(\bmod p)\\y_1&\equiv(a_1+xb_1)(\bmod q)\\y_2&\equiv(a_2+xb_2)(\bmod q)\end{aligned}$$

因为 α 是 G 的生成元，所以存在唯一的指数 $c_1,c_2,a_0\in Z_q$ 使得 $\gamma_1\equiv\alpha^{c_1}(\bmod p),\gamma_2\equiv\alpha^{c_2}(\bmod p),\beta\equiv\alpha^{a_0}(\bmod p)$。因此，上面的同余式组等价于下列的同余式组：

$$\begin{aligned}c_1&\equiv a_1+a_0a_2(\bmod q)\\c_2&\equiv b_1+a_0b_2(\bmod q)\\y_1&\equiv a_1+xb_1(\bmod q)\\y_2&\equiv a_2+xb_2(\bmod q)\end{aligned}$$

易知这个方程组关于变量 a_1,a_2,b_1,b_2 在 Z_q 上的系数矩阵的秩为 3。由假定条件知，该方程组在 Z_q 上至少有一个解，该解可由 K 获得。所以该方程组的解空间的维数为 $4-3=1$，

故恰有 q 个解。

事实 3：假定 K 是一个密钥，$y=\mathrm{Sig}_K(x)$，$\mathrm{Ver}_K(x',y')=$真，$x'\neq x$，那么至多有一个等价于 K 的密钥 K' 使得 $y=\mathrm{Sig}_{K'}(x)$，且 $y'=\mathrm{Sig}_{K'}(x')$。

事实 3 的证明类似于事实 2 的证明。

事实 2 和事实 3 表明，给定消息 x 的一个合法签名 y，有 q 个可能的密钥将 x 签为 y。但对任何消息 $x'\neq x$，这 q 个密钥将对 x' 产生 q 个不同的签名。

综上所述，我们有下述定理。

定理 7.5.1 给定 $\mathrm{Sig}_K(x)=y$ 和 $x'\neq x$，敌手 O 只能以 $1/q$ 的概率计算 $\mathrm{Sig}_K(x')$。

定理 7.5.1 不依赖于 O 的计算能力，因为 O 无法区别出 B 使用了这 q 个可能的密钥中的哪一个，所以安全性是无条件的。迄今为止，我们已经说明了给定消息 x 的一个签名 y，O 不能计算 B 对不同的消息 x' 的签名 y'。但 O 也许能计算一个可通过验证的伪造的签名 $y''\neq\mathrm{Sig}_K(x')$。然而，如果给 B 一个合法的伪造签名，那么他能以概率 $1-1/q$ 产生一个伪造证明(proof of forgery)。伪造证明是值 $a_0=\log_\alpha\beta$，a_0 只有可信中心知道。

现在假定 B 拥有一对 (x',y'') 使得 $\mathrm{Ver}(x',y'')=$真，并且 $y''\neq\mathrm{Sig}_K(x')$。也就是说，$\gamma_1\gamma_2^{x'}\equiv\alpha^{y_1''}\beta^{y_2''}(\mathrm{mod}\ p)$，$y''=(y_1'',y_2'')$。B 自己能计算 x' 的签名 $y'=(y_1',y_2')$，并且 $\gamma_1\gamma_2^{x'}\equiv\alpha^{y_1'}\beta^{y_2'}(\mathrm{mod}\ p)$。因此，$\alpha^{y_1''}\beta^{y_2''}\equiv\alpha^{y_1'}\beta^{y_2'}(\mathrm{mod}\ p)$。设 $\beta=\alpha^{a_0}\mathrm{mod}\ p$，则上式等价于 $\alpha^{y_1''+a_0y_2''}\equiv\alpha^{y_1'+a_0y_2'}(\mathrm{mod}\ p)$或 $y_1''+a_0y_2''\equiv y_1'+a_0y_2'(\mathrm{mod}\ q)$，即 $y_1''-y_1'\equiv a_0(y_2'-y_2'')(\mathrm{mod}\ q)$。因为 y''是一个伪造，所以 $y_2'\not\equiv y_2''(\mathrm{mod}\ q)$，所以 $(y_1'-y_2'')^{-1}\mathrm{mod}\ q$ 存在，从而

$$a_0=\log_\alpha\beta=(y_1''-y_1')(y_1'-y_2'')^{-1}\mathrm{mod}\ q$$

当然，是在假定签名者 B 自己不能计算离散对数 $\log_\alpha\beta$ 的计算假设下，第三方才能接收 B 的这一伪造证明。

最后，我们指出，Van Heyst-Pedersen 签名方案是一个一次签名方案，因为如果使用同一个密钥 K 签两个不同的消息，那么很容易计算出密钥 K。

7.6 群数字签名方案和盲数字签名方案

7.6.1 群数字签名方案

由 Chaum 和 Van Heijst 提出的群数字签名(group digital signature)[27]允许群中的各个成员以群的名义匿名地签发消息。群数字签名是具有下列三个特性的一种数字签名：

(1) 只有群的成员才能代表那个群签发消息；

(2) 签名的接收者能验证它是那个群的一个合法签名，但不能揭示它是群中的哪一个成员产生的；

(3) 在后来发生争端的情况下，借助于群成员或一个可信的机构能识别出那个签名者。

一个群数字签名方案主要由三个算法即签名算法、验证算法和识别算法(识别算法主要用来识别签名者)组成。文献[27]给出了群数字签名方案的一些具体实现，在这些方案中，每个成员 U_i 拥有一个秘密密钥 S_i，使用这些 S_i 中的任何一个签发消息，签名能使用群的公钥 K 来验证。若发生争端，通过识别算法可找出真正的签名者。关于群签名的进一

步研究可参阅文献[28,29,30]。

这种签名的一个实用的例子是投标，群——所有的提交投标的公司组成的集合，成员——每个公司。每个公司匿名地使用群数字签名签他的投标，当特定的投标被选中后，那个签名者能被识别出，而所有的别的投标的签名者仍然是匿名的。如果签名者反悔他的投标，那么无需签名者的合作，他的身份能被计算出。

与群数字签名相关的两个概念是群定向数字签名[31]和多重数字签名[32]。群定向数字签名允许群的某些子集代表那个群签名，但它没有提供识别签名者的方法。多重数字签名要求由许多人来签署一个消息。

下面我们来描述一个群数字签名方案，该方案是 Chaum 和 Van Heijst 于 1991 年在文献[27]中提出的四个方案中的第一个方案。首先我们来描述该方案的基本思想。假定有 n 个人构成了一个群 G，T 是一个可信中心，可信中心给每个人分发一张秘密密钥表（这些表是互不相交的），并且将所有群成员的所有秘密密钥所对应的公钥以一个随机的次序排成一张表公开。这样，每个人就能用他自己的秘密密钥表中的秘密密钥签署消息，并且接收者能用公开的表中的对应公钥验证签名。每个密钥只能使用一次，否则接收者能将这些签名联系在一起。因为是可信中心 T 产生的秘密密钥表，所以 T 知道公钥和签名者之间的对应关系，一旦发生争端，T 就可解决这一争端。在这个方案中，每个人得到一张固定的表，方案的公钥的长度是群成员人数 n 的一个线性函数，但每个人所能签的消息的量是固定的。

设 p 是一个使得在 Z_p 上计算离散对数是不可行的大素数，g 是 Z_p 的一个生成元。设每一个群成员只有一个秘密密钥 $s_i(1\leqslant i\leqslant n)$，对应的公钥是 $g^{s_i}\bmod p(1\leqslant i\leqslant n)$，可信中心 T 有一张这些公钥和群成员的名字相对应的表。可信中心 T 对每一个群成员 i 随机选择一个数 $r_i\in Z_p^*$（这里的 r_i 也称作盲因子）并将表 $(g^{s_i})^{r_i}(1\leqslant i\leqslant n)$ 公开（这种公钥也称作盲公钥(blinded public key)。每个群成员 i 可使用 $s_ir_i\bmod(p-1)$ 作为秘密密钥，使用 ElGamal 型数字签名方案、不可否认的数字签名方案等数字签名方案对消息 x 进行签名。可信中心 T 可周期地给群成员更换盲因子 $r_i(1\leqslant i\leqslant n)$。

该方案的签名过程和验证过程与所选用的数字签名方案有关。当接收者收到一个消息的签名时，他用公开的公钥表中的每一个公钥去验证，只要有一个公钥使签名通过验证，就说明这个签名是该群的一个合法签名。如果发生争端，接收者可将通过验证的公钥和签名一起送交给可信中心 T，T 可利用这些信息及他存贮的公钥和群成员的名字的对应表找出相对应的签名者。

这个方案的一个突出优点是可信中心 T 不能伪造签名，而且每个成员只有一个真正的秘密密钥。只有在一个周期段内的同一个签名者的签名才能被联系在一起。如果用户 i 在 t 个周期段内使用的 r_i 都被泄露，那么用户 i 在这 t 个周期段的签名都能被联系在一起，但没有泄露秘密密钥 s_i 的任何信息。这说明签名者的秘密性可被无条件地保护。这个方案的缺点是如果有新的群成员参加，所有的群成员都不得不改变他们的密钥，否则接收者能将旧的群成员和新的群成员区别开来。

7.6.2 盲数字签名方案

盲数字签名(blind digital signature)[33]在需要实现某些参加者的匿名性的密码协议

中有着广泛而重要的应用，诸如在选举协议、安全的电子支付系统中等。盲数字签名方案是具有下列两个特性的一种数字签名方案：(1)消息的内容对签名者是盲的(不可见的)；(2)在签名被接收者泄露后，签名者不能追踪签名。目前已有大量的文献讨论了盲数字签名方案的实现和应用问题。关于盲数字签名方案的应用将在第11章详细介绍。本节我们只介绍两种类型的盲数字签名方案，即基于RSA体制的盲数字签名方案和基于离散对数问题的盲数字签名方案。

盲数字签名在签名时，接收者首先将被签的消息进行盲变换，把变换后的消息(称为盲消息)发送给签名者，签名者对盲消息进行签名并把消息送还给接收者，接收者对签名再做逆盲变换，得出的消息即为原消息的盲签名。这个过程如图7.6.1所示。

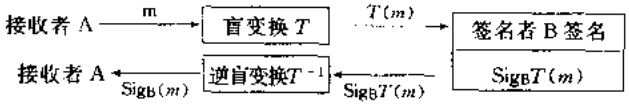


图 7.6.1　盲数字签名方案的签名过程

盲变换实现了盲数字签名的特性(1)。要实现盲数字签名的特性(2)必须要求当签名者后来看到盲签名时不能与盲消息联系起来。

首先，我们来描述一个基于RSA体制的盲数字签名方案。

签名者B选择两个大素数 p 和 q，以及一个单向函数 f，并随机选择一个 e，使得 $\gcd(e,\varphi(n))=1$，其中 $n=pq$。由 $ed\equiv 1(\bmod \varphi(n))$ 求出 $d\equiv e^{-1}\bmod \varphi(n)$。B公开 n，e 和 f，保密 p，q 和 d。

如果接收者A想让B给他盲签一个消息 x，那么A先随机选择一个数 $k\in Z_n^*$ (k 称为盲因子)，计算 $x'\leftarrow f(x)k^e \bmod n$，并将 x' 发送给B。B收到 x' 进行签名得 $y'=\mathrm{Sig}_B(x')=(x')^d \bmod n$。然后B将签名 y' 发送给A，A计算

$$y = y'/k \bmod n = (f(x)k^e)^d/k \bmod n = f^d(x) \bmod n$$

现在A得到了B对 x 的一个盲签名 y。显然，y 是 x 的一个合法签名。但B在签名过程中从来没有看到过 x 和 y，他无法将 (x,y) 和 (x',y') 联系起来。

其次，我们来描述两个基于离散对数问题的盲数字签名方案。

方案一：签名者选择两个大素数 p 和 q，使得在 Z_p 上计算离散对数是困难的，并且 $q|(p-1)$。另外，选择一个阶为 q 的数 $\alpha\in Z_p^*$ 和一个秘密密钥 x，公开 p，q，α 和 $y=\alpha^x \bmod p$，保密 x。

如果接收者A想让签名者B给他盲签一个消息 m，那么B先随机选择一个数 $\tilde{k}\in Z_q^*$，计算 $\tilde{r}=\alpha^{\tilde{k}} \bmod p$，并将 $\tilde{r}$ 发送给A。A随机选择两个数 $a,b\in Z_q$，计算 $r=\tilde{r}^a\alpha^b \bmod p$，$\tilde{m}=am\tilde{r}r^{-1} \bmod q$，并将 $\tilde{m}$ 发送给B。B计算 $\tilde{s}=(x\tilde{r}+\tilde{k}\tilde{m}) \bmod q$，并将 $\tilde{s}$ 发送给A。A计算 $s=(\tilde{s}r\tilde{r}^{-1}+bm) \bmod q$，则B对消息 m 的签名是一对整数 (r,s)，验证方程是 $\alpha^s=y^r r^m \bmod p$。

发送者A　　签名者B

$\tilde{k}\in Z_q^*$

$\xleftarrow{\tilde{r}}$

$\tilde{r}=\alpha^{\tilde{k}} \bmod p$

$a,b\in Z_q$

$r=\tilde{r}^a a^b \bmod p$ $\xrightarrow{\tilde{m}}$ $\tilde{s}=(x\tilde{r}+\tilde{k}\tilde{m})\bmod q$

$\tilde{m}=amr\tilde{r}^{-1}\bmod q$

$s=(\tilde{s}r\tilde{r}^{-1}+bm)\bmod q$ $\xleftarrow{\tilde{s}}$

方案二：除了多选一个自由碰撞的单向 Hash 函数 H 外，系统的其他参数的选取与方案一相同。如果接收者 A 想让签名者 B 给他盲签一个消息 m，那么 A 随机选择一个数 $t\in Z_q^*$，计算 $m_0=m^t$，并将 m_0 发送给 B，要求 B 对 m_0 签名。签名者 B 随机选择一个数 $s\in Z_q$，计算 $a_0=a^s$，$b_0=m_0^s$ 和 $z_0=m_0^r$，并将 (a_0,b_0,z_0) 发送给接收者 A。接收者 A 随机选择 $u\in Z_q^*$ 和 $v\in Z_q$，计算 $a=(a_0g^v)^u$ 和 $b=(b_0^{1/t}m^v)^u$。A 同时也计算 $Z=Z_0^{1/t}$，$c=H(m,Z,a,b)$ 和盲口令 $c_0=c/u \bmod q$，并将 c_0 发送给 B。签名者 B 发送回一个回答 $r_0=s+c_0x$。接收者 A 接收这个口令，当且仅当 $g^{r_0}=a_0y^{c_0}$ 且 $m_0^{r_0}=b_0Z_0^{c_0}$。接收者 A 计算 $r=(r_0+v)u \bmod q$，则 $\delta=(Z,a,b,r)$ 是 m 的一个盲签名。验证方程是 $g^r=ay^c$ 和 $m^r=bZ^c$。

可证明[56]，如果接收者 A 采纳上述协议，那么签名者没有得到 m 和 δ 的任何信息。方案二的一种推广了的情况可参见文献[57]。

盲数字签名在某种程度上是保护了参加者的利益，但不幸的是，盲数字签名的匿名性能被犯罪分子滥用；为了阻止这种滥用文献[33]引入了公平盲数字签名的概念并给出了一些具体实现。公平盲数字签名比盲数字签名多了一个特性：借助于可信中心可将消息-签名对和签名者在签名协议中观察到的对应的盲消息联系起来。也就是说，通过可信中心，签名者可追踪签名，感兴趣的读者请参阅文献[33,34]。

7.7 注记和文献

目前已提出了大量的数字签名方案[1,6,35,36]，除了本章介绍的和提到的之外，还有 Esign 数字签名方案[37,38]、基于椭圆曲线的数字签名方案[8]、指定验证者的数字签名方案[39]、共享验证数字签名方案[40,41,42,43,44]、具有恢复消息功能的数字签名方案[9]、基于有限自动机的数字签名方案[45,46]、基于身份的数字签名方案[47,48]、以及由识别协议转化而来的数字签名方案，诸如 Schnorr 数字签名方案、Okamoto 数字签名方案、Guillon-Quisquater 数字签名方案、FFS 数字签名方案、FS 数字签名方案等等。关于由识别协议转化而来的数字签名方案将在第 9 章介绍。

关于纠错、加密和数字签名相结合的方案，感兴趣的读者请参阅文献[49,50,51]。

参 考 文 献

[1] Mitchell, C, J., Piper, F. and Wild, P., Digital Signatures, In: Contemporary Cryptology, The Science of Information Integrity, IEEE Press, 1992, pp. 325—378.

[2] Rivest, R. L., Shamir, A. and Adleman, L., A Method for Obtaining Digital Signatures and Public-key Cryptosystem, Comm. ACM Vol. 2(2), pp. 120—146, Feb. 1978.

[3] Elgamal, L., A Public Key Cryptosystem and a Signature Scheme Base on Discrete Logarithm, IEEE Transactions on Information Theory, 31(1985), pp. 469—472.

[4] Chaum, D. and Roijakkers, S., Unconditionall secure Digital Signatures, Advances in Cryptology-Crypto'90, Berlin: Springer-Verlag, 1991, pp. 206－214.

[5] Diffie, W. and Hellman, M., New Directions in Cryptography, IEEE Trans, Inform. Theory, 1976, Vol. IT-22 (6), pp. 644－654.

[6] 冯登国,数字签名技术概述,通信保密,No. 3, 1996,15－22 页.

[7] Stinson, D. R., Cryptography theory and Practice, CRC Press, 1995.

[8] Menezes, A. J., Elliptic Curve Public Key Cryptosystems, Kluwer Academic Publishers, 1993.

[9] Nyberg, K. and R. A., Message Recovery for Signature Schemes Based on the Discrete Logarithm Problem, Advances in Cryptology Eurocrypt'94, Springer-Verlag, 1995, pp. 182－193.

[10] Schnorr, C. P., Efficient Signature Generation by Smart Cards, J. Cryptology, 4, 1991, pp. 161－174.

[11] FIPS PUB XX, 1993 February 1, Digital Signature Standard.

[12] Smid, M. E. and Branstad, D. K., Response to Comments on the NIST Proposed Digital Signature Standard, Advances in Cryptology——Crypto'92, Springer-Verlag, 1993, pp. 76－88.

[13] Bleichenbacher, D., Generating ElGamal Signatures Without Knowing the Secret Key, Advances in Cryptology-Eurocrypt'96, Springer-Verlag, 1996, pp. 10－18.

[14] Vaudenay, S., Hidden Collisions on DSS, Advances in Cryptology-Crypto'96, Springer-Verlag, 1996, pp. 83－88.

[15] Kocher, P., Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, Advances in Cryptology-Crypto'96, Springer-Verlag, 1996, pp. 104－113.

[16] Simmons, G. J., Subliminal Communication Is Easy Uning the DSA, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, 1994, pp. 218－232.

[17] Naccache, D., Mraihi, D., Vaudenay, S. and Raphaeli, D., Can D. S. A. Be Improved? Complexity Trade-offs with the Digital Signature Standard, Advances in Cryptology-Eurocrypt'94, Springer-Verlag, 1995, pp. 77－85.

[18] Gennaro, R., Jarecki, S., Krawczyk, H. and Rabin, T., Robust Threshold DSS Signatures, Advances in Cryptology-Eurocrypt'96, Springer Verlag, 1996, pp. 354 371.

[19] Bos, J. N. E. and Chaum, D., Provably Unforgeable Signatures, Advances in Cryptology-Crypto'92, Springer-Verlag, 1993, pp. 1－14.

[20] Chaum, D. and Van Antwerpen, H., Undeniable Signatures, Advances in Cryptology——Crypto'89, Springer-Verlag, 1990, pp. 212－216.

[21] Chaum, D., Van Heijst, E. and Pfitzmann, B., Cryptographically Strong Undeniable Signatures, Uncorditionally Secure for the Signer, Advances in Cryptology——Crypto'91, Springer-Verlag, 1992, pp. 470－484.

[22] Chaum, D., Zero-knowledge Undeniable Signatures, Advances in Cryptology——Eurocrypt'90, Springer-Verlag, 1991, pp. 458－464.

[23] Boyar, J., Chaum, D. and Damgard, I., Convertible Undeniable Signatures, Advances in Cryptology——Crypto'90, Springer-Verlag, 1991, pp. 189－205.

[24] Damgard, I. B. and Pedersen, T. P., New Convertible Undeniable Signature Scheme, Advances in Cryptology——Eurocrypt'96, Springer-Verlag, 1996, pp. 372－386.

[25] Waidner, M. and Pfitzman, B., The Dining Cryptographets in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serriceability, Advances in Cryptology——Eurocrypt'89, Springer-Verlag, 1990, p. 690.

[26] Van Heyst, E. and Pedersen, T. P., How to Make Efficient Fail-stop Signatures, Advances in Cryptology——Eurocrypt'92, Springer-Verlag, 1993, pp. 366－377.

[27] Chaum, D. and Van Heijst, E., Group Signatures, Advances in Cryptology——Eurocrypt'91, Springer-Verlag, 1991, pp. 257－265.

[28] Chen, L. and Pedersen, T. P., New Group Signatures Schemes, Advances in Cryptology——Eurocrypt'94, Springer-Verlag, 1995, pp. 171－181.

[29] Chen, L. and Pedersen, T. P., On the Efficiency of Group Signature Providing Information—— Theoretic

Anonymity, Advances in Cryptology——Eurocrypt'95, Springer-Verlag, 1995, pp. 39－49.

[30] Kim, S. J., Park, S. J. and Won, D. H., Convertible Group Signatures, Advances in Cryptology——Asiacrypt'96, Springer-Verlag, 1996, pp. 311－321.

[31] Desmedt, Y., Threshold Cryptosystems, Advances in Cryptology-Auscrypt'92, Springer-Verlag, 1993, pp. 3－14.

[32] Boyd, C., Digital Multisignatures, In: Cryptography and Coding, Clarendon Press, 1989, pp. 241－246.

[33] Stadler, M., Piveteau, J. M. and Camenisch, J., Fair Blind Signatures, Advances in Cryptology——Eurocrypt'95, Springer-Verlga, pp. 209－219.

[34] 冯登国，基于离散对数问题的盲数字签名，通信保密，No. 1，1997.

[35] Schneier, B., Applied Cryptography——Protocols, Algorithms, and Source Code in C, John Wiley & Sons, Inc., 1993.

[36] Menezes, A. J., Van Oorschot, P. C. and Vanstone, S. A., Handbook of Applied Cryptography, IEEE Press, 1996.

[37] Okamoto, T. and Shiraishi, A., A Fast Signature Scheme Based on Quadratic Inequalities, Proceedings of the 1985 Symposium on Security and Privacy, IEEE, Apr. 1985, pp. 123－132.

[38] Okamoto, T., Provably Secure and Practical Indentification Schemes and Corresponding Signature Schemes, Advances in Cryptology——Crypto'92, Springer-Verlag, 1993, pp. 31－53.

[39] Chaum, D., The Designated——Confirmer Signatures, Advances in Cryptology——Eurocrypt'94, Springer-Verlag, 1995, pp. 86－91.

[40] Desmedt, Y. And Frankel, Y., Shared Generation of Authenticators and Signatures, Advances in Cryptology——Crypto'91, Springer-Verlag, 1992, pp. 457－469.

[41] Harn, L., (t,n) Threshold Signature and Digital Multisignature, Workshop on Cryptography and Data Security, 1993, pp. 61－73.

[42] Li, C. M., Hwang, T. and Lee, N., Threshold-multisignature Schemes Where Suspected Forgery Implies Traceability of Adversarial Shareholders, Advances in Cryptology——Eurocrypt'94, Springer-Verlag, 1995, pp. 194－204.

[43] Lu Langru, Zhao Renjie, Hou Lining, A (t, n) Threshold Group Signature Scheme, 密码学进展——Chinacrypt'96, 1996, 177－184 页.

[44] 冯登国，基于离散对数问题的(t,n)门限数字签名方案，密码与信息，No. 1，1997，11－13 页.

[45] 陶仁骥、陈世华，一种有限自动机公开钥密码体制和数字签名，计算机学报，1985，No. 8，401—409 页.

[46] 陶仁骥、陈世华，基于身份的密码体制和数字签名的有限自动机公开钥密码实现，密码学进展——Chinacrypt'92，科学出版社，北京，1992，87－104 页.

[47] Shamir, A., Identity-based Cryptosystems and Signature Schemes, Advances in Cryptology-Crypto'84, Springer-Verlag, 1985, pp. 47－53.

[48] Harn, L. and Yang, S. B., ID-based Cryptographic Schemes for User Identification, Digital Signature, and Key Distribution, IEEE J. Select Areas Commun, Vol. 71, pp. 757－760.

[49] 王新梅，纠错码数字签名，加密纠错公钥体制，电子学报，19(5)，1991，48—54 页.

[50] Wang, X. M., A Digital Signature Scheme Constructed with Error-correcting Codes, IEE electronics letters, 26 (13), 1990, pp. 898－899.

[51] 冯登国、肖国镇，一种数字签名、加密和纠错公钥体制，科技通报，11(3)，1995，156－158 页.

[52] 王育民、何大可，保密学——基础与应用，西安电子科技大学出版社，西安，1990.

[53] Feng, D. G., Verifiable Signature Sharing for the DSA with Heuristic Secrecy, IEE Electronics Letters, Vol. 32, No. 15, 1996, pp. 1570－1571.

[54] 冯登国，一种新型的数字签名方案，计算机工程与应用，Vol. 34，No. 5，1998，36—37 页.

[55] Goldwasser, S., Micali, S. and Rivest, R. L., A Digital Signature Scheme Secure Against Adaptive Chosen-message Attacks, SIAM Journal on Computing, 1988, pp. 281－308.

[56] Chaum, D., Pedersen, T. P., Wallet Databases with Observers, Advances in Cryptology-Crypto'92, Springer-

Verlag, 1993, pp. 89—105.
[57] Brands, S., Untraceable Off-line Cash in Wallets with Observers, Advances in Cryptology-Crypto'93, Springer Verlag, 1994, pp. 302—318.

第 8 章　杂凑(Hash)函数

我们在第 7 章中曾介绍了 RSA 数字签名方案的一些弱点，并指出可用杂凑(Hash)函数来克服这些弱点。密码学上的杂凑函数(也称杂凑算法或方案)是一种将任意长度的消息压缩到某一固定长度的消息摘要(message digest)的函数。将杂凑函数应用到数字签名中可带来下列一些好处：

(1) 可破坏数字签名方案的某种数学结构，诸如同态结构。

(2) 可提高数字签名的速度。当签名者想签一个消息 x 时，他首先构造一个消息摘要 $z=h(x)$(h 是一个杂凑函数)，然后计算签名 $y=\mathrm{Sig}_K(z)$。使用杂凑函数的数字签名方案参见图 8.0.1。

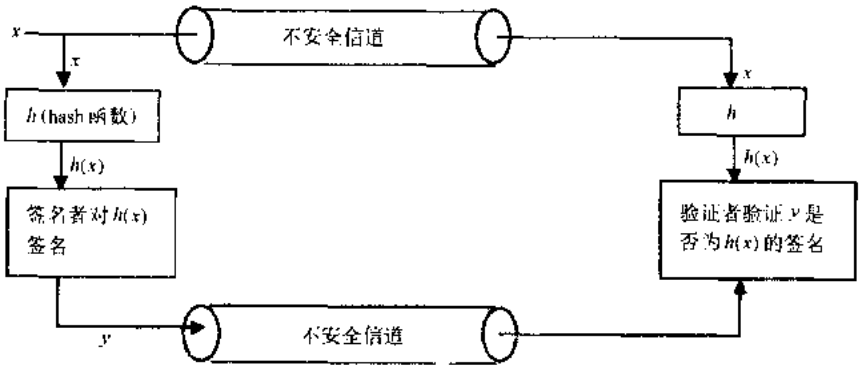


图 8.0.1　使用 Hash 函数的数字签名方案

(3) 无需泄露签名所对应的消息，可将签名泄露，比如对消息 x 的签名是 $y=\mathrm{Sig}_K(z)$，其中 $z=h(x)$，可将 (z,y) 公开，而保密 x。

(4) 可将签名变换和加密变换分开来，允许用私钥密码体制实现保密，而用公钥密码体制实现数字签名。在 ISO 的公开系统互联参考模型(open system interconnect reference model)中，这种分离的一个优点是可在不同层提供消息的完整性(integrity)和机密性(confidentality)。

杂凑函数除了可用于数字签名方案中之外，还可用于其它方面，诸如消息的完整性检测、消息的起源认证检测等。例如，文件的所有者为了保障一个文件的完整性，即阻止对文件的非法改变，他首先必须获得一个文件的 Hash 值，自己保存这个文件的 Hash 值的一份拷贝，然后将文件存贮在可公开的地方。每当他使用文件时，他计算存贮的文件的 Hash 值，然后与他自己保存的拷贝进行比较，如果相等，文件是完整的，没有被改动。否则，文件已被改动。在通信中使用 Hash 函数进行完整性检测的基本模型如图 8.0.2 所示。

从理论上来讲，安全的 Hash 函数的存在性依赖于单向函数的存在性。文献[1]讨论了如何利用一个给定的单向函数来构造一个安全的 Hash 函数。同时，文献[1]也对 1993 年之前人们在 Hash 函数方面所做的工作做了一个比较全面的总结。

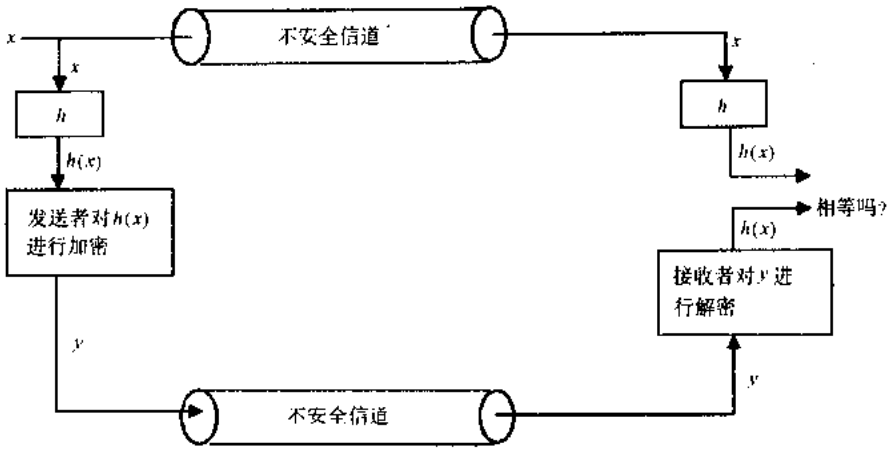


图 8.0.2 使用 Hash 函数进行完整性检测的基本模型

目前已设计出了大量的 Hash 函数，诸如 Rabin Hash 方案[2]、Merkle Hash 方案[3]、N-Hash 算法[4,5]、MD_4 算法[6]、MD_5 算法[7]、SHA[8]等等。还可在文献[1,9,10]中找到其他一些 Hash 函数。限于篇幅，本章我们只能介绍一些有代表性的 Hash 算法，想全面了解现有的 Hash 算法的读者请参阅文献[1,9,10]。

8.1 Hash 函数的分类

根据 Hash 函数的安全水平人们将 Hash 函数分成两大类：一类是强碰撞自由的 Hash 函数(strong collision-free hash functions)；另一类是弱碰撞自由的 Hash 函数(weak collision-free hash functions)。

一个强碰撞自由的 Hash 函数是满足下列条件的一个函数 h：

(1) h 的输入可以是任意长度的任何消息或文件 M；

(2) h 的输出的长度是固定的(该长度必须足够长以抵抗所谓的生日攻击，根据今天的计算技术和能力，至少应为 128 比特长)；

(3) 给定 h 和 M，计算 $h(M)$是容易的；

(4) 给定 h 的描述，找两个不同的消息 M_1 和 M_2，使得 $h(M_1)=h(M_2)$是计算上不可行的(如果有两个消息 $M_1, M_2, M_1 \neq M_2$，使得 $h(M_1)=h(M_2)$，我们就说这两个消息是碰撞消息或这两个消息碰撞)。

一个强碰撞自由的 Hash 函数暗含着单向性[12]，所以强碰撞自由的 Hash 函数又称强单向 Hash 函数。

一个弱碰撞自由的 Hash 函数与一个强碰撞自由的 Hash 函数的前三个条件(1)～(3)完全一样，不同的只是第(4)个条件。在一个弱碰撞自由的 Hash 函数中，(4)给定 h 的描述和一个随机选择的消息 M_1，找另一个消息 $M_2, M_2 \neq M_1$，使得 $h(M_1)=h(M_2)$是计算上不可行的。

显然，强碰撞自由的 Hash 函数要比弱碰撞自由的 Hash 函数的安全性强。一个弱碰撞自由的 Hash 函数不能保证找不到一对消息 $M_1, M_2, M_1 \neq M_2$，使得 $h(M_1)=h(M_2)$。这

说明也许有消息 $M_1, M_2, M_1 \neq M_2$，使得 $h(M_1)=h(M_2)$。然而，对随机选择的消息 M_1，要故意地选择另一个消息 $M_2, M_2 \neq M_1$，使得 $h(M_1)=h(M_2)$是计算上不可行的。

在数字签名中，应用强或弱碰撞自由的 Hash 函数可阻止签名的伪造或抵赖。这一点可由条件(4)来保证。

为了加强弱碰撞自由的 Hash 函数的安全性，人们建议在弱碰撞自由的 Hash 函数中引入随机性。引入随机性的方法主要有二种：第一种方法是用一个好的分组密码使用一个真正随机的密钥加密消息使消息随机化。随机密钥也将级联到导致的密文的开头[11]。第二种方法是在对消息进行杂凑之前，给消息随机选择一个前缀，这样的一个随机前缀将有效地随机化 Hash 值。第二种方法是从一族 Hash 函数中随机选择 Hash 函数而不是随机化消息本身[12]。

值得注意的是弱碰撞自由的 Hash 函数随着重复使用次数的增加而安全性逐渐降低，这是因为用同一个弱碰撞自由的 Hash 函数杂凑的消息越多，找到一个消息的 Hash 值等于先前的消息的 Hash 值的机会就越大，从而系统的总体安全性降低，而强碰撞自由的 Hash 函数不会因其重复使用而降低安全性[11]。

根据方案是否需要使用秘密密钥来控制可将 Hash 函数分成两类，即带秘密密钥的 Hash 函数和不带秘密密钥的 Hash 函数。只有在消息中存在多余消息，我们才能将真正的消息和虚假的消息区别开来。带秘密密钥的 Hash 函数和不带秘密密钥的 Hash 函数是在消息中引进可控多余性的两种基本方法。在带秘密密钥的 Hash 函数中，多余消息是消息 M 的 Hash 函数值，此时的 Hash 函数是由只有通信双方知道的一个秘密密钥 K 来控制。在这种情况下，多余消息称为消息认证码(message authentication code 或 MAC)。ANST9.9 消息认证标准[14]和 ISO9797[15]中就使用了这种技术。一般来讲，MAC 使用私钥密码算法诸如 DES 和一个秘密认证密钥，该密钥用于确保只有授权的人才能产生具有合适的 MAC 的消息。例如，设 A 和 B 是通信的双方，A 可使用 DES 的 CBC 模式产生一个 MAC。设初始向量 $IV=0$，明文组为 $x_1, x_2, \cdots, x_n$。A 用 DES 的 CBC 模式使用密钥 K 构造密文组 $y_1, y_2, \cdots, y_n$，将 y_n 定义为消息(明文)组 $x_1, x_2, \cdots, x_n$ 的 MAC。A 将明文组 $x_1, x_2, \cdots, x_n$ 连同 MAC 发送给 B。当 B 收到 $x_1, x_2, \cdots, x_n$ 时，他能使用秘密密钥 K 计算 $y_1, y_2, \cdots, y_n$，并验证 y_n 是否和他收到的 MAC 一致，从而达到认证的目的。有时需要同时实现加密和认证。这可通过下列办法来完成：A 首先使用秘密密钥 K_1 产生 $x_1, x_2, \cdots, x_n$ 的一个 MAC。然后将 x_{n+1} 定义为 MAC，并且使用另一个秘密密钥 K_2 加密 $x_1, x_2, \cdots, x_{n+1}$ 产生 $y_1, y_2, \cdots, y_{n+1}$，将 $y_1, y_2, \cdots, y_{n+1}$ 发送给 B。当 B 收到 $y_1, y_2, \cdots, y_{n+1}$ 时，他首先使用 K_2 解密，然后使用 K_1 检测 x_{n+1} 是 $x_1, x_2, \cdots, x_n$ 的一个 MAC。也可以用另一种办法来同时完成加密和认证：A 首先使用 K_1 加密 $x_1, x_2, \cdots, x_n$，获得 $y_1, y_2, \cdots, y_n$。然后使用 K_2 产生 $y_1, y_2, \cdots, y_n$ 的一个 MAC y_{n+1}，并将 $y_1, y_2, \cdots, y_n, y_{n+1}$ 发送给 B。当 B 收到 $y_1, y_2, \cdots, y_n, y_{n+1}$ 时，B 将使用 K_2 验证 MAC，然后使用 K_1 解密 $y_1, y_2, \cdots, y_n$。在一个安全的 MAC 方案中，拥有秘密密钥的人不能找到一个碰撞的消息；否则，方案不能解决通信双方之间出现的争端。这种方法的缺点是增加了密钥管理的负担。在不带秘密密钥的 Hash 函数中，多余消息仅仅是消息 M 的一个 Hash 函数值，无需使用一个秘密密钥。在这种情况下，多余消息称为窜改检测码(manipulation deetection code 或 MDC)。安全的 Hash 标准(SHS)就采用了这种技术。因为产生 MDC 的 Hash 函数是公开知道的，所以为了阻止攻击者成功

地代替攻击，消息 M 连同 MDC 通常需要加密。MDC 的优点是只需要公开知道的成分，因此，它简化了安全消息处理系统中的密钥管理，而且认证可与加密函数及其工作模式分离开来，在 OSI 参考模型中，加密和消息认证可在不同的协议层实现。这种方法的缺点是如果保密机制的安全性受到危及，那么完整性就没有保障。从这里我们也可以体会到保密和认证是两个独立的概念。

根据 Hash 函数自身的结构，考虑它是否用一个分组密码来提供一个单向函数可将 Hash 函数分为两大类，即基于分组密码的 Hash 函数(block-cipher-based Hash functions)和基于非分组密码的 Hash 函数(non-block-cipher-based Hash functions)。文献[1]按这种分类介绍了大量的 Hash 函数。

我们通常所说的 Hash 函数的安全性在无特别声明的条件下是指：在现有的计算资源(包括时间、空间、资金等)下，找到一个碰撞是不可行的。

8.2 Hash 函数的延拓准则

现在我们提出这样一个问题：如果我们手里边有一个具有有限的定义域的强碰撞自由的 Hash 函数，那么我们如何将它延拓为一个具有无限的定义域的强碰撞自由的 Hash 函数呢？本节我们主要来回答这个问题。

设 $h: F_2^m \to F_2^t$ 是一个强碰撞自由的 Hash 函数，$m \geqslant t+1$。下面我们使用 h 来构造一个强碰撞自由的 Hash 函数 $h^*: X \to F_2^t$，这里

$$X = \bigcup_{i=m}^{\infty} F_2^i$$

首先，我们考虑 $m \geqslant t+2$ 的情况。

我们将 X 中的元素看成比特串。用 $|x|$ 表示 x 的长度，$x \| y$ 表示比特串 x 和 y 的级联。假定 $|x|=n>m$，可将 x 表示为 $x=x_1 \| x_2 \| \cdots \| x_k$，这里 $|x_1|=|x_2|=\cdots=|x_{k-1}|=m-t-1$，$|x_k|=m-t-1-d$，$0 \leqslant d \leqslant m-t-2$。因此，我们有 $k=\left[\frac{n}{m-t-1}\right]$，$[x]$ 表示不小于 x 的最小整数。

现在我们根据下列方法用 h 来定义 $h^*(m \geqslant t+2)$：

(1) 对 $i=1$ 到 $k-1$ 置 $y_i=x_i$；

(2) $y_k=x_k \| 0^d$(给 x_k 的右边级联 d 个 0 使得 $|y_k|=m-t-1$)；

(3) 设 y_{k+1} 是 d 的二元表示(给 d 的二元表示的左边级联 0 使得 $|y_{k+1}|=m-t-1$)；

(4) $g_1=h(0^{t+1} \| y_1)$；

(5)对 $i=1$ 到 k 计算 $g_{i+1}=h(g_i \| 1 \| y_{i+1})$；

(6) $h^*(x)=g_{k+1}$。

记 $y(x)=y_1 \| y_2 \| \cdots \| y_{k+1}$。为了杂凑 x，我们首先构造 $y(x)$，然后处理组 $y_1, y_2, \cdots, y_{k+1}$。显然，每当 $x \neq x'$ 时，$y(x) \neq y(x')$，这一点很重要。

下面的定理 8.2.1 告诉我们：如果 h 是安全的，那么 h^* 也是安全的。

定理 8.2.1 假定 $h: F_2^m \to F_2^t$ 是一个强碰撞自由的 Hash 函数，$m \geqslant t+2$，则上述定义的函数 $h^*: \bigcup_{i=m}^{\infty} F_2^i \to F_2^t$ 是一个强碰撞自由的 Hash 函数。

证明：假定我们能找到$x\neq x'$，使得$h^*(x)=h^*(x')$。现在我们说明如何利用这一对x和x'在多项式时间内找到h的一个碰撞。因为h被假定是强碰撞自由的，所以我们就推出了一个矛盾，从而说明h^*是强碰撞自由的。

记$y(x)=y_1\|y_2\|\cdots\|y_{k+1}$，$y(x')=y'_1\|y'_2\|\cdots\|y'_{l+1}$，这里$x$和$x'$在第(2)步分别被级联了$d$和$d'$个0。记在第(4)步和第(5)步所计算的结果分别为$g_1,\cdots,g_{k+1}$和$g'_1,\cdots,g'_{l+1}$。

下面我们根据是否$|x|\equiv|x'|(\mathrm{mod}(m-t-1))$来分两种情况讨论。

情况1：$|x|\not\equiv|x'|(\mathrm{mod}(m-t-1))$，此时，$d\neq d'$，从而$y_{k+1}\neq y'_{l+1}$。而

$$h(g_k\|1\|y_{k+1})=g_{k+1}=h^*(x)=h^*(x')=g'_{l+1}=h(g'_l\|1\|y'_{l+1})$$

所以$g_k\|1\|y_{k+1}$和$g'_k\|1\|y'_{l+1}$是h的一对碰撞消息。

情况2：$|x|\equiv|x'|(\mathrm{mod}(m-t-1))$。对这种情况根据是否有$|x|=|x'|$再分两种情况讨论。

(1) $|x|=|x'|$，此时$k=l$，$y_{k+1}=y'_{k+1}$。因为

$$h(g_k\|1\|y_{k+1})=g_{k+1}=h^*(x)=h^*(x')=g'_{k+1}=h(g'_k\|1\|y'_{k+1})$$

所以如果$g_k\neq g'_k$，那么$g_k\|1\|y_{k+1}$和$g'_k\|1\|y'_{k+1}$就是h的一对碰撞消息。如果$g_k=g'_k$，那么

$$h(g_{k-1}\|1\|y_k)=g_k=g'_k=h(g'_{k-1}\|1\|y'_k)$$

如果$g_{k-1}\neq g'_{k-1}$或$y_k\neq y'_k$，我们就找到了h的一个碰撞，否则我们继续往后退，直到最后我们有

$$h(0^{t+1}\|y_1)=g_1=g'_1=h(0^{t+1}\|y'_1)$$

如果$y_1\neq y'_1$，那么我们找到了h的一个碰撞。可是不可能对所有的i，$1\leqslant i\leqslant k+1$，都有$y_i=y'_i$，这是因为如果对所有的$i$，$1\leqslant i\leqslant k+1$，有$y_i=y'_i$，则$y(x)=y(x')$，这暗含着$x=x'$，但我们假定$x\neq x'$，这就产生了矛盾。所以我们总能找到$h$的一个碰撞。

(2) $|x|\neq|x'|$，不失一般性，可假定$|x'|>|x|$，从而$l>k$。我们首先按类似于(1)中的方式去找h的碰撞，如果我们按这种方式没有找到h的碰撞，那么此时必有

$$h(0^{t+1}\|y_1)=g_1=g'_{l-k+1}=h(g'_{l-k}\|1\|y'_{l-k+1})$$

但$0^{t+1}\|y_1$的第$t+1$个比特是0，而$g'_{l-1}\|1\|y'_{l-k+1}$的第$t+1$个比特是1，所以$0^{t+1}\|y_1$和$g'_{l-1}\|1\|y'_{l-k+1}$就是h的一对碰撞消息。

综上所述，如果h^*有一个碰撞，那么我们可以找到h的一个碰撞。这就证明了我们所期望的结论。

其次，我们考虑$m=t+1$的情况。在这种情况下的构造方法与$m\geqslant t+2$的情况下的构造方法不同。

假定$|x|=n>m$，$x=x_1x_2\cdots x_n$。我们首先使用一种特殊的方法编码x。按下述方式定义函数f：$f(0)=0$，$f(1)=01$。其次根据下列方法用h来定义$h^*(m=t+1)$：

(1) 设$y=y_1y_2\cdots y_n=11\|f(x_1)\|f(x_2)\|\cdots\|f(x_n)$；

(2) $g_1=h(0^t\|y_1)$；

(3) 对$i=1$到$k-1$计算　　$g_{i+1}=h(g_i\|y_{i+1})$；

(4) $h^*(x)=g_k$。

定义在第(1)步的编码规则：$x \mapsto y = y(x)$满足下述两个性质：

(a) 如果 $x \neq x'$，那么 $y(x) \neq y(x')$；

(b) 不存在两个串 $x \neq x'$ 和一个串 Z 使得 $y(x) = Z \| y(x')$，即没有编码是另一个编码的后缀(这是因为每一个编码以 11 开始，而其余的串不存在两个连续的 1)。

下面的定理 8.2.2 告诉我们，如果 h 是安全的，那么 h^* 也是安全的。

定理 8.2.2 假定 $h: F_2^{t+1} \to F_2^t$ 是一个强碰撞自由的 Hash 函数，则上述定义的 $h^*: \bigcup_{i=t+1}^{\infty} F_2^i \to F_2^t$ 是一个强碰撞自由的 Hash 函数。

证明：假定我们能找到 x 和 x'，$x \neq x'$ 使得 $h^*(x) = h^*(x')$，记 $y(x) = y_1 y_2 \cdots y_k$，$y(x') = y_1' y_2' \cdots y_l'$，我们分两种情况来讨论。

情况 1：$k = l$。像定理 8.2.1 的证明过程一样，我们或者能找到 h 的一个碰撞或者获得 $y = y'$，但是 $y = y'$ 暗含着 $x = x'$，这就产生了矛盾。

情况 2：$k \neq l$。不失一般性，假定 $l > k$，我们利用类似于情况 1 的处理方式寻找 h 的碰撞。假定我们没有找到 h 的一个碰撞，此时必有下列一串等式成立：

$$\begin{aligned} y_k &= y_l' \\ y_{k-1} &= y_{l-1}' \\ &\vdots \\ y_1 &= y_{l-k+1}' \end{aligned}$$

但这与编码规则 $x \mapsto y(x)$ 的性质(b)矛盾，所以一定可找到 h 的一个碰撞。

综上所述，h^* 是一个强碰撞自由的 Hash 函数。

将上述两个定理可合并为一个定理如下。

定理 8.2.3 假定 $h: F_2^m \to F_2^t$ 是一个强碰撞自由的 Hash 函数，$m \geqslant t+1$，则一定存在一个强碰撞自由的 Hash 函数 $h^*: \bigcup_{i=m}^{\infty} F_2^i \to F_2^t$。在 $h^*(x)$ 的计算中，应用 h 的次数至多为：

$$\begin{cases} 1 + \left[\dfrac{n}{m-t-1}\right] & m \geqslant t+2 \\ 2n+2 & m = t+1 \end{cases}$$

这里 $|x| = n$。

上述介绍的方法在文献[17]中称作串联方法，而在文献[18]中称作 meta 方法。为了并行计算 Hash 值，文献[17]基于上述方法给出了一种变化的构造，称作并行方法或树方法。感兴趣的读者请参阅文献[17]。

8.3 Hash 函数的攻击方法

评价 Hash 方案的一个最好的方法是看一下一个敌手找到一对碰撞消息所花的代价有多高。一般地，假设敌手知道 Hash 算法。敌手的主要攻击目标是找到一对或更多对碰撞消息。目前已有一些攻击 Hash 方案和计算碰撞消息的方法。这些方法中的一些是一般的方法，可用于攻击任何类型的 Hash 方案，诸如生日攻击(birthday attack)。而另一些是特殊的方法，只能用于攻击某些特殊类型的 Hash 方案，诸如适用于攻击具有分组链结构的 Hash 方案的中间相遇攻击(meet-in-the-middle attack)，适用于攻击基于模算术的

Hash 函数的修正分组攻击(correcting block attack)。

8.3.1 生日攻击

生日攻击方法没有利用 Hash 函数的结构和任何代数弱性质，它只依赖于消息摘要的长度，即 Hash 值的长度。这种攻击对 Hash 函数提出了一个必要的安全条件，即消息摘要必须足够的长。生日攻击这个术语来自于所谓的生日问题：在一个教室中最少应有多少学生才使得至少有两个学生的生日在同一天的概率不小于 1/2？

这个问题的答案是 23。下面我们来详细描述生日攻击方法。

设 $h: X \to Y$ 是一个 Hash 函数，X 和 Y 都是有限的，并且 $|X| \geqslant 2|Y|$，记 $|X|=m$，$|Y|=n$。显然至少有 n 个碰撞，问题是如何去找这些碰撞。一个很自然的方法是随机选择 k 个不同的元素 $x_1, x_2, \cdots, x_k \in X$，计算 $y_i=h(x_i)$，$1 \leqslant i \leqslant k$，然后确定是否有一个碰撞发生。这个过程类似于把 k 个球随机地扔到 n 个箱子里边，然后检查看是否某一箱子里边至少有两个球。k 个球对应于 k 个随机数 $x_1, x_2, \cdots, x_k$，n 个箱子对应于 Y 中的 n 个可能的元素。我们将计算用这种方法找到一个碰撞的概率的下界，该下界只依赖于 k 和 n，而不依赖于 m。

因为我们感兴趣的是碰撞概率的下界，所以我们将假定对所有的 $y \in Y$，有 $|h^{-1}(y)| \approx m/n$。这个假定是合理的，这是因为如果原像集 $h^{-1}(y)(y \in Y)$ 不是近似相等的，那么找到一个碰撞的概率将增大。

因为原像集 $h^{-1}(y)(y \in Y)$ 的个数都近似相等，并且 $x_i(1 \leqslant i \leqslant k)$ 是随机选择的，所以可将 $y_i=h(x_i)(1 \leqslant i \leqslant k)$ 视作 Y 中的随机元素（$y_i(1 \leqslant i \leqslant k)$ 未必不同）。但计算 k 个随机元素 $y_1, y_2, \cdots, y_k \in Y$ 是不同的概率是一件容易的事情。依次考虑 $y_1, y_2, \cdots, y_k$。y_1 可任意地选择；$y_2 \neq y_1$ 的概率为 $\frac{n-1}{n}=1-1/n$；$y_3 \neq y_1, y_2$ 的概率为 $\frac{n-2}{n}=1-\frac{2}{n}$；$\cdots$；$y_k \neq y_1, y_2, \cdots, y_{k-1}$ 的概率为 $\frac{n-(k-1)}{n}=1-\frac{k-1}{n}$。

因此，没有碰撞的概率是

$$\left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{k-1}{n}\right)=\prod_{i=1}^{k-1}\left(1-\frac{i}{n}\right)$$

如果 x 是一个比较小的实数，那么 $1-x \approx e^{-x}$，这个估计可由下式推出：

$$e^{-x}=1-x+\frac{x^2}{2!}-\frac{x^3}{3!}+\cdots$$

现在我们来估计没有碰撞的概率

$$\prod_{i=1}^{k-1}\left(1-\frac{i}{n}\right) \approx \prod_{i=1}^{k-1} e^{-\frac{i}{n}}=e^{-\frac{k(k-1)}{2n}}$$

所以至少有一个碰撞的概率约为 $1-e^{-\frac{k(k-1)}{2n}}$。我们设 ε 是至少有一个碰撞的概率，则 $\varepsilon \approx 1-e^{-\frac{k(k-1)}{2n}}$，从而有 $k^2-k \approx n \ln \frac{1}{(1-\varepsilon)^2}$。去掉 $-k$ 这一项，我们有 $k^2 \approx n \ln \frac{1}{(1-\varepsilon)^2}$，即 $k \approx \sqrt{2n \ln \frac{1}{1-\varepsilon}}$。

如果我们取 $\varepsilon=0.5$，那么 $k \approx 1.17\sqrt{n}$。这表明，仅杂凑 $\sqrt{n}$ 个 X 的随机的元素就能以 50%的概率产生一个碰撞。注意 ε 的不同选择将导致一个不同的常数因子，但 k 与 $\sqrt{n}$

仍成正比例。

如果 X 是一个教室中所有学生的集合，Y 是一个非润年的 365 天的集合，$h(x)$表示学生 x 的生日，现在我们来处理生日问题。这时 $n=365$，$\varepsilon=0.5$，由 $k\approx1.17\sqrt{n}$ 可知，$k\approx22.3$。因此，前面提到的生日问题的答案为 23。

生日攻击隐含着消息摘要的长度的一个下界。一个 40 比特长的消息摘要是很不安全的，因为仅仅用 2^{20}(大约一百万)次随机杂凑可至少以 1/2 的概率找到一个碰撞。为了抵抗生日攻击，通常建议消息摘要的长度至少应选取为 128 比特，此时生日攻击需要约 2^{64} 次杂凑。毫无疑问，安全 Hash 标准的输出长度选为 160 比特是出于这种考虑。

生日攻击的一种变形是中间相遇攻击，如何用中间相遇攻击来攻击数字签名方案可参阅文献[19]。

8.3.2 特殊攻击

一些攻击方法不像生日攻击一样可用于攻击任何类型的 Hash 函数，它们只能用于攻击某些特殊类型的 Hash 函数，本小节作一简单介绍。

8.3.2.1 中间相遇攻击

中间相遇攻击是生日攻击的一种变形，它不是比较 Hash 值，而是比较链中的中间变量。这种攻击主要适用于攻击具有分组链结构的 Hash 方案。中间相遇攻击的基本观点是：攻击者将消息分成两部分，对伪造消息的第一部分他从初始值开始逐步向中间阶段产生 r_1 个变量；对伪造消息的第二部分他从 Hash 结果开始逐步退回中间阶段产生 r_2 个变量。在中间阶段有一个匹配的概率与生日攻击成功的概率一样，详细论述请参阅文献[20]。一个方案可通过下述办法来避免中间相遇攻击：使方案是不可逆的，这一点在文献[21]中已被注意到。

为了避免中间相遇攻击，文献[22]和[23]考虑了二重迭代方案。然而，文献[24]和[25]把中间相遇攻击推广到一般的中间相遇攻击，这种攻击不仅能破译二重迭代方案，而且也能破译 p-重迭代方案，所需的运算量仅为 $O(10^p\cdot 2^{\frac{n}{2}})$，这里 n 是 Hash 值的长度。

8.3.2.2 修正分组攻击和差分分析

在修正分组攻击中，为了修正杂凑结果并获得期望的值，伪造消息和一个分组级联。这种攻击通常应用于最后一个组，因此也称为修正最后分组攻击(correcting last block attack)。当然，这种攻击可应用于任何别的组。应用这种攻击方法分析 Hash 函数的一些具体实例参见文献[26,27]。

在第 5 章我们已经详细介绍了差分分析的基本思想。差分分析是攻击分组密码的一种方法。这种攻击也可用来攻击某些 Hash 算法，诸如 Merkle Hash 方案、N-halsh 函数、MD_5 等，详细分析过程参见文献[28,29]。

另外，针对 Hash 算法的一些弱点可对 Hash 算法进行攻击。诸如可利用 Hash 方案的代数结构对方案进行一些攻击，诸如组的插入、置换和代替等攻击。文献[30]利用 Hash 函数的代数结构及其所使用的分组密码的弱点说明了如何来攻击一些 Hash 方案，诸如 DES 的一些众所周知的弱点(即互补性、(半)弱密钥、密钥碰撞等)可用来攻击基于 DES

的 Hash 方案。

8.4 Hash 函数的构造

文献[16]将构造 Hash 函数的方法分为三种，这种分类方法类似于 Rueppel 将流密码的构造分为四种的做法。第一种方法是信息论方法，它以信息论为基础，且它提供了无条件安全性，即安全性与敌手的计算能力无关。关于无条件安全认证的第一个结果出现在文献[31]中，后来由 Simmons 发展了这个理论[32]；关于这种理论的研究和发展在本书第 2 章中我们已作过一些讨论。第二种方法是复杂性理论方法，这种方法起源于计算的抽象模型，并假定敌手具有有限的计算能力，它只能提供计算上的安全性。第三种方法是系统论方法，其安全性估计是以所知道的攻击该系统的最好的算法和必需的计算能力或实现算法的专用硬件的实际估计为基础的。这是一种很实际的方法。本节我们没有按照文献[16]中的分类介绍 Hash 函数的构造。我们将根据 Hash 函数的安全基础即安全假设来介绍一些有代表性的 Hash 函数。

8.4.1 一个基于离散对数问题的 Hash 函数

基于一些困难问题诸如离散对数问题、因子分解问题、背包问题等可构造出一些 Hash 函数，利用这种方法所构造的 Hash 函数的安全性依赖于这些问题的困难性。本小节我们介绍一个基于离散对数问题的 Hash 函数，称为 Chaum-van Heijst-Pfitzmann Hash 函数[33]。这个 Hash 函数在实际使用中不是很快，但是在合理的假设下，可证明它是安全的。其详细构造过程如下：

假定 p 是一个大素数，并且 $q=(p-1)/2$ 也是一个素数。设 α 和 β 是 Z_p 的两个本原元。值 $\log_\alpha\beta$ 不是公开的，并假定计算 $\log_\alpha\beta$ 是计算上不可行的。

Hash 函数 $h:\{0,1,\cdots,q-1\}\times\{0,1,\cdots,q-1\}\to Z_p\backslash\{0\}$ 定义为

$$h(x_1,x_2)=\alpha^{x_1}\beta^{x_2}\bmod p$$

下面我们来证明 h 是一个强碰撞自由的 Hash 函数。如果我们证明了 h 是一个强碰撞自由的 Hash 函数，那么由 8.2 节中介绍的 Hash 函数的延拓方法可将 h 延拓为具有无限的定义域的 Hash 函数。

定理 8.4.1 假定给定 Chaum-Van Heijst-Pfitzmann Hash 函数 h 的一个碰撞，那么离散对数 $\log_\alpha\beta$ 能被有效地计算。

证明：假定 $(x_1,x_2),(x_3,x_4),(x_1,x_2)\neq(x_3,x_4)$，是 h 的一对碰撞消息，即 $h(x_1,x_2)=h(x_3,x_4)$，那么我们有 $\alpha^{x_1}\beta^{x_2}\equiv\alpha^{x_3}\beta^{x_4}\bmod p$，即 $\alpha^{x_1-x_3}\equiv\beta^{x_4-x_2}(\bmod p)$。记 $d=\gcd(x_4-x_2,p-1)$。因为 $p-1=2q$，且 q 是一个素数，所以 $d\in\{1,2,q,p-1\}$。因此，d 有四种取值情况，下面将分四种情况来分别讨论。

情况 1：$d=1$。此时 x_4-x_2 关于模 $p-1$ 有逆，设 $y=(x_4-x_2)^{-1}\bmod(p-1)$，则 $\beta\equiv\beta^{(x_4-x_2)y}(\bmod p)\equiv\alpha^{(x_1-x_3)y}(\bmod p)$，所以 $\log_\alpha\beta=(x_1-x_3)(x_4-x_2)^{-1}\bmod(p-1)$。

情况 2：$d=2$。因为 $p-1=2q$，q 为奇数，所以 $\gcd(x_4-x_2,q)=1$。设 $y=(x_4-x_2)^{-1}\bmod q$，此时 $(x_4-x_2)y=kq+1$（对某一整数 k），则

$$\beta^{(x_4-x_2)y}\equiv\beta^{kq+1}(\bmod p)\equiv(-1)^k\beta(\bmod p)\equiv\pm\beta(\bmod p)$$

因为 $\beta^q \equiv -1 \pmod{p}$，所以

$$\alpha^{(x_4-x_2)y} \equiv \beta^{(x_1-x_3)y} \pmod{p} \equiv \pm\beta \pmod{p}$$

故 $\log_\alpha \beta = (x_1-x_3)y \bmod (p-1)$ 或 $\log_\alpha \beta = (x_1-x_3)y+q \bmod (p-1)$

易检查出这两个等式中的哪一个成立。因此，像 $d=1$ 的情况一样，我们可计算 $\log_\alpha \beta$。

情况 3：$d=q$。因为 $0 \leqslant x_1 \leqslant q-1, 0 \leqslant x_3 \leqslant q-1$，所以 $-(q-1) \leqslant x_4-x_2 \leqslant q-1$。因此 $\gcd(x_4-x_2, p-1)=q$ 是不可能的。换句话说，这种情况不出现。

情况 4：$d=p-1$。这种情况只有在 $x_2=x_4$ 的情况下可能发生。但此时有 $\alpha^{x_1} \equiv \alpha^{x_3} \pmod{p}$，所以 $x_1=x_3$，这样 $(x_1,x_2)=(x_3,x_4)$ 就产生了矛盾，故这种情况也不出现。

综上所述，在 Z_p 上计算离散对数 $\log_\alpha \beta$ 是计算上不可行的假设下，Hash 函数 h 是强碰撞自由的。

8.4.2 基于私钥密码算法的 Hash 函数

虽然我们可证明 Chaum-Van Heijst-Pfitzmann Hash 函数是强碰撞自由的，但这种 Hash 函数速度太慢，在实际中是不实用的。本小节我们来介绍使用已有的私钥密码算法构造 Hash 函数的方法。

设 (P,C,K,ϵ,D) 是一个计算上安全的私钥密码算法。为讨论方便起见，让我们假定 $P=C=K=F_2^n$。为了阻止生日攻击，这里我们假定 $n \geqslant 128$。这个要求就排除了使用 DES（事实上，DES 的密钥长度不等于明文长度）。

设 x 是一个串，通过某种填充方式可将 x 填充成长度为 n 的倍数的串。不失一般性，假定 $x=x_1 \| x_2 \| \cdots \| x_k, x_i \in F_2^n, 1 \leqslant i \leqslant k$。

基本的观点是从一个固定的初始值 $g_0=IV$ 开始，根据下列规则依次构造 $g_1, g_2, \cdots, g_k$：

$$g_i = f(x_i, g_{i-1})$$

这里 f 是根据取定的密码体制的加密函数所确定的一个函数。然后将消息摘要定义为 $h(x)=g_k$。

这种类型的一些 Hash 函数已被提出，但它们中的许多已被表明是不安全的（独立于所使用的基础密码体制的安全性）。然而，下列四种这种类型的方案似乎还是安全的：

(1) $g_i = E_{g_{i-1}}(x_i) \oplus x_i$；

(2) $g_i = E_{g_{i-1}}(x_i) \oplus x_i \oplus g_{i-1}$；

(3) $g_i = E_{g_{i-1}}(x_i \oplus g_{i-1}) \oplus x_i$；

(4) $g_i = E_{g_{i-1}}(x_i \oplus g_{i-1}) \oplus x_i \oplus g_{i-1}$。

8.4.3 直接构造法

Rivest 于 1990 年设计了一个称为 MD_4 的 Hash 函数[6]，该 Hash 函数的设计没有基于任何假设和密码体制，这种直接构作法受到人们的广泛青睐。这种 Hash 函数的速度很快，非常实用。MD_4 Hash 函数公布不久，den Boer 和 Bosselaers[34]以及 Merke 发现，如果剥掉 MD_4 算法的第一轮或最后一轮是不安全的，但这些分析并没有扩展到整个算法上。为了增强安全性和克服 MD_4 的缺陷 Rivest 于 1991 年对 MD_4 作了六点改进，将改进的算法称作 MD_5[7]。MD_5 公布后，文献[35,36]对它进行了分析；文献[36]中指出，MD_5 的压缩函

数 G 有一个碰撞，虽然这并不是说 MD_5 是不安全的，但这与 MD_5 的基本设计准则之一即设计一个无碰撞的压缩函数相违背，这不得不引起使用者的担心。

本小节我们首先来详细描述 MD_4 Hash 算法，其次讨论 MD_4 和 MD_5 的不同之处。MD 代表消息摘要(message digest)。

给定一个串 x，我们首先将产生一个下列形式的串：

$$M = M_{[0]}M_{[1]}\cdots M_{[N-1]}$$

这里每个 $M_{[i]}(0\leqslant i\leqslant N-1)$是长为 32 比特的串，$N\equiv 0(\mathrm{mod}\ 16)$。我们将每个 $M_{[i]}$称为一个字(word)。由 x 产生 M 的算法如下：

(1) $d=447-(|x| \bmod 512)$(当 $d<0$ 时，按模 512 处理)；

(2) 设 l 表示 $|x| \bmod 2^{64}$的二元表示，$|l|=64$；

(3) $M=x\parallel 1\parallel 0^d\parallel l$。

在 M 的构造中，首先在 x 的右边填充一个 1，然后级联足够多的 0，使得整个消息的长度模 512 为 448，最后再级联 $|x|\mathrm{mod}2^{64}$的二元表示，长度为 64，产生的 M 的长度为 512 的倍数，所以这时可将 M 分成 32-比特字，字的个数 N 是 16 的倍数。

现在我们从 M 开始构造一个 128 比特长的消息摘要，其构造过程如下：

(1) 给四个寄存器 A,B,C,D 赋初始值(用十六进制表示)：

$A=67452301$

$B=$efcdab89

$C=$98badcfe

$D=10325476$

(2) for $i=0$ to $N/16-1$ do；

(3) for $j=0$ to 15 do，

$X_{[j]}=M_{[16i+j]}$

(4) 将四个寄存器 A、B、C、D 的值存贮在另外四个寄存器 AA，BB，CC，DD 之中，

AA=A　BB=B　CC=C　DD=D

(5) 执行第一轮；

(6) 执行第二轮；

(7) 执行第三轮；

(8) A=A+AA　B=B+BB　C=C+CC　D=D+DD。

每次处理 16 个字，这 16 个字的处理过程从第(3)步到第(8)步。最后一次即第 $N/16$ 次结束时，寄存器 A、B、C、D 的值的级联作为 Hash 值(消息摘要)输出，长度为 128 比特。

下面我们将对上述构作过程中涉及到的一些运算以及三个轮作进一步的描述。

设 X 和 Y 表示输入的字，下列描述一些运算：

(1) $X\wedge Y$　　表示 X 与 Y 按位逻辑“与(and)”；

(2) $X\vee Y$　　表示 X 与 Y 按位逻辑“或(or)”；

(3) $X\oplus Y$　　表示 X 与 Y 按位逻辑“异或(xor)”；

(4) $\overline{X}$　　表示 X 的按位逻辑“补(complement)”；

(5) $X+Y$　　表示整数模 2^{32}加法运算；

(6) $X\lll s$　　表示将 X 循环左移 s 个位置$(0\leqslant s\leqslant 31)$。

上述运算都是很快的，只有模 2^{32} 加法运算是算术运算。如果要真正在计算机上实现 MD_4，为了正确地完成加法运算，必须考虑运行它的计算机的基础结构。假定 $a_1a_2a_3a_4$ 是一个字的 4 个字节，每个字节是 8 比特。我们把每个 a_i 视为 0 到 255 之间的一个整数的二元表示。

在一个 big-endian 结构中(诸如一个 sun SPARCstation)，这个字表示整数 $a_1 2^{24}+a_2 2^{16}+a_3 2^8+a_4$。在一个 little-endion 结构中(诸如 Intel 80XXX line)，这个字表示整数 $a_4 2^{24}+a_3 2^{16}+a_2 2^8+a_1$。$MD_4$ 的设计是以 Little-endian 结构为基础的，但消息摘要独立于基础结构。所以如果我们想在一个 big-endian 计算机上运行 MD_4，我们必须按下述办法来完成加法运算 $X+Y$：

(1) 交换 x_1 和 x_4，x_2 和 x_3，y_1 和 y_4，y_2 和 y_3；

(2) 计算 $Z=X+Y \bmod 2^{32}$；

(3) 交换 z_1 和 z_4，z_2 和 z_3。

MD_4 的第一、二、三轮分别使用了函数 f、g 和 h。它们分别定义为：

$$f(X,Y,Z)=(X \wedge Y) \vee (\overline{X} \wedge Z)$$

$$g(X,Y,Z)=(X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$$

$$h(X,Y,Z)=X \oplus Y \oplus Z$$

MD_4 中的三个轮是不同的，它不像 DES 中的十六轮都是相同的。下面我们来分别描述这三个轮。

第一轮：

(1) for $k=0$ to 3 do;

(2) $A=(A+f(B,C,D)+X_{[4k]})\lll 3$;

(3) $D=(D+f(A,B,C)+X_{[4k+1]})\lll 7$;

(4) $C=(C+f(D,A,B)+X_{[4k+2]})\lll 11$;

(5) $B=(B+f(C,D,A)+X_{[4k+3]})\lll 19$.

第二轮：

(1) $A=(A+g(B,C,D)+X_{[0]}+5a827999)\lll 3$;

(2) $D=(D+g(A,B,C)+X_{[4]}+5a827999)\lll 5$;

(3) $C=(C+g(D,A,B)+X_{[8]}+5a827999)\lll 9$;

(4) $B=(B+g(C,D,A)+X_{[12]}+5a827999)\lll 13$;

(5) $A=(A+g(B,C,D)+X_{[1]}+5a827999)\lll 3$;

(6) $D=(D+g(A,B,C)+X_{[5]}+5a827999)\lll 5$;

(7) $C=(C+g(D,A,B)+X_{[9]}+5a827999)\lll 9$;

(8) $B=(B+g(C,D,A)+X_{[13]}+5a827999)\lll 13$;

(9) $A=(A+g(B,C,D)+X_{[2]}+5a827999)\lll 3$;

(10) $D=(D+g(A,B,C)+X_{[6]}+5a827999)\lll 5$;

(11) $C=(C+g(D,A,B)+X_{[10]}+5a827999)\lll 9$;

(12) $B=(B+g(C,D,A)+X_{[14]}+5a827999)\lll 13$;

(13) $A=(A+g(B,C,D)+X_{[3]}+5a827999)\lll 3$;

(14) $D=(D+g(A,B,C)+X_{[7]}+5a827999)\lll 5$;

(15) $C=(C+g(D,A,B)+X_{[11]}+5a827999)\ll 9$;

(16) $B=(B+g(C,D,A)+X_{[15]}+5a827999)\ll 13$;

第三轮：

(1) $A=(A+h(B,C,D)+X_{[0]}+6ed9eba1)\ll 3$;

(2) $D=(D+h(A,B,C)+X_{[8]}+6ed9eba1)\ll 9$;

(3) $C=(C+h(D,A,B)+X_{[4]}+6ed9eba1)\ll 11$;

(4) $B=(B+h(C,D,A)+X_{[12]}+6ed9eba1)\ll 15$;

(5) $A=(A+h(B,C,D)+X_{[2]}+6ed9eba1)\ll 3$;

(6) $D=(D+h(A,B,C)+X_{[10]}+6ed9eba1)\ll 9$;

(7) $C=(C+h(D,A,B)+X_{[6]}+6ed9eba1)\ll 11$;

(8) $B=(B+h(C,D,A)+X_{[14]}+6ed9eba1)\ll 15$;

(9) $A=(A+h(B,C,D)+X_{[1]}+6ed9eba1)\ll 3$;

(10) $D=(D+h(A,B,C)+X_{[9]}+6ed9eba1)\ll 9$;

(11) $C=(C+h(D,A,B)+X_{[5]}+6ed9eba1)\ll 11$;

(12) $B=(B+h(C,D,A)+X_{[13]}+6ed9eba1)\ll 15$;

(13) $A=(A+h(B,C,D)+X_{[3]}+6ed9eba1)\ll 3$;

(14) $D=(D+h(A,B,C)+X_{[11]}+6ed9eba1)\ll 9$;

(15) $C=(C+h(D,A,B)+X_{[7]}+6ed9eba1)\ll 11$;

(16) $B=(B+h(C,D,A)+X_{[15]}+6ed9eba1)\ll 15$;

MD_4 的速度很快，在 Sun SPARCstations 上软件实现的速度达到 1.4 兆字节/秒。因为 MD_4 的安全性不基于任何困难问题，诸如因子分解问题、离散对数问题，所以该方案的安全性要想得到人们的信任，像 DES 一样只能随时间来考验。目前还没有有效的算法能攻破该方案。

MD_5 是 MD_4 的改进形式，它比 MD_4 更复杂，但设计思想相似并且也产生 128 比特的 Hash 值。MD_5 使用了四轮而不是三轮，运行速度比 MD_4 慢了大约 30%（MD_5 在 Sum SPARCstations 上软件实现的速度大约为 0.9 兆字节/秒）。MD_5 与 MD_4 相比，主要作了以下六点改进：

(1) 增加了第四轮，第四轮所使用的函数为 $I(X,Y,Z)=(X\vee\bar{Z})\oplus Y$;

(2) 第二轮的函数 g 从 $(X\wedge Y)\vee(X\wedge Z)\vee(Y\wedge Z)$ 变为 $(X\wedge Z)\vee(Y\wedge\bar{Z})$，以减弱它的对称性；

(3) 进入第二轮和第三轮的输入字的次序被改变；

(4) 每一轮中的移位数已改变，并且各轮中的移位数互不相同；

(5) 每一步有一个唯一的加法常数；

(6) 每一步加上了上一步的结果，这样会产生更好的“雪崩效应”。

前四点改进是为了抵抗文献[34]中的攻击，而后两点改进是为了加强 MD_5 的安全性。文献[37]对 MD_5 作了进一步的改进，称为 Haval Hash 函数，它有 8 个 32 比特的初始值（也称链接变量），是 MD_5 的两倍，它的轮数可变，从 3 轮到 5 轮（每轮有 16 步），并且能产生长度为 128,160,192,224 或 256 比特的 Hash 值。Haval 比 MD_5 快，三轮快 60%，四轮快 15%。Haval 可抵抗文献[36]中对 MD_5 的攻击方法。

8.5 安全 Hash 标准(SHS)

美国国家标准技术研究所(NIST)和美国国家安全局(NSA)一道设计了与美国数字签名标准(DSS)一起使用的安全 Hash 算法(SHA)[8](标准是安全 Hash 标准,SHA 是用于标准的算法)。当然,该算法不仅仅局限于应用到数字签名中,它也可以应用到任何需要 Hash 算法的地方。安全 Hash 标准(SHS)于 1992 年 1 月 31 日在联邦记录中公布,1993 年 5 月 11 日起采纳为标准。1994 年 7 月 11 日作了一个小小的修改,1995 年 4 月 17 日公布了修改了的版本,在新的版本中将 SHA 称为 SHA-1,SHS 的设计原则与 MD_4Hash 函数的设计原则极其相似,它是 MD_4 的一种变形,它比 MD_4 的速度要慢大约 0.2 兆字节/秒(在一个 Sun SAPRCstation 上),它的设计是以 big-endion 结构为基础的。但 SHS 的设计者没有详细公开 SHS 的设计决策。本节以最新版本为准介绍 SHA-1。

输入消息 x 的长度 $|x|$ 限制为 $<2^{64}$ 比特,SHA-1 的输出的长度为 160 比特,即 5 个字。

给定一个消息 x,$|x|<2^{64}$,首先我们将产生一个长度是 512 的倍数的串:$M=M_{[0]}M_{[1]}\cdots M_{[N-1]}$。产生办法与 MD_4 的产生办法完全一样。这里 $M_{[i]}(0\leqslant i\leqslant N-1)$都是长为 32 比特的串即一个字,$N\equiv 0(\mathrm{mod}\ 16)$。

现在从 M 开始构造一个 160 比特的消息摘要(与 MD_4 不同,MD_4 的输出是 128 比特),其构造过程如下:

(1) 给五个寄存器 A、B、C、D、E 赋初始值:

A=67452301

B=efcdab89

C=98badcfe

D=10325476

E=c2d2e1f0

(2) for $i=0$ to $N/16-1$ do;

(3) for $j=0$ to 15 do

$X_{[j]}=M_{[16i+j]}$

(4) 将给定的 16 个字 $X_{[0]},X_{[1]},\cdots,X_{[15]}$ 扩展成 80 个字,这一步计算其它 64 个字(MD_4 中没有这一扩展,它是与 MD_4 的主要区别):

$$\text{for } t=16 \text{ to } 79 \text{ do}$$

$$X_{[t]}=(X_{[t-3]}\oplus X_{[t-8]}\oplus X_{[t-14]}\oplus X_{[t-16]})\ll 1$$

(称 $X_{[t]}=(X_{[t-3]}\oplus X_{[t-8]}\oplus X_{[t-14]}\oplus X_{[t-16]})\ll 1$ 为扩展函数;在 1993 年的版本中,扩展函数为 $X_{[t]}=X_{[t-3]}\oplus X_{[t-8]}\oplus X_{[t-14]}\oplus X_{[t-16]}$,在新版本中作了一个修改,加了"$\ll 1$")。

(5) 将 5 个寄存器 A、B、C、D、E 中的值存贮在另外 5 个寄存器 AA、BB、CC、DD、EE 之中,

AA=A　　BB=B　　CC=C　　DD=D　　EE=E

(6) 执行第一轮;

(7) 执行第二轮;

(8) 执行第三轮；

(9) 执行第四轮；

(10) A=A+AA　　B=B+BB　　C=C+CC　　D=D+DD　　E=E+EE

每次处理16个字，这16个字的处理过程从第(3)步到第(10)步。最后一次即第$N/16$次结束时的寄存器A、B、C、D、E的值的级联作为Hash值输出，长度为160比特。

SHA-1用到的运算与MD_4中的运算完全一样。下面将对SHA-1的四个轮作进一步的描述。

第一、二、三、四轮所使用的函数分别为：

$f_1(X,Y,Z)=(X\wedge Y)\vee(\overline{X}\wedge Z)$

$f_2(X,Y,Z)=X\oplus Y\oplus Z$

$f_3(X,Y,Z)=(X\wedge Y)\vee(X\wedge Z)\vee(Y\wedge Z)$

$f_4(X,Y,Z)=f_2(X,Y,Z)$

第一、二、三、四所使用的固定的常数分别为：

$K_1=$5a827999

$K_2=$6ed9eba1

$K_3=$8f1bbcdc

$K_4=$ca62c1d6

第一轮：

for $k=0$ to 19 do

TEMP$=(A\ll 5)+f_1(B,C,D)+E+X_{[k]}+K_1$

$E=D$　　$D=C$　　$C=(B\ll 30)$　　$B=A$　　$A=$TEMP。

第二轮：

for $k=20$ to 39 do

TEMP$=(A\ll 5)+f_2(B,C,D)+E+X_{[k]}+K_2$

$E=D$　　$D=C$　　$C=(B\ll 30)$　　$B=A$　　$A=$TEMP。

第三轮：

for $k=40$ to 59 do

TEMP$=(A\ll 5)+f_3(B,C,D)+E+X_{[k]}+K_3$

$E=D$　　$D=C$　　$C=(B\ll 30)$　　$B=A$　　$A=$TEMP。

第四轮：

for $k=60$ to 79 do

TEMP$=(A\ll 5)+f_4(B,C,D)+E+X_{[k]}+K_4$

$E=D$　　$D=C$　　$C=(B\ll 30)$　　$B=A$　　$A=$TEMP。

与MD_4相比，SHA-1增加了一轮，每轮作20次操作，而且在SHA-1中每一步都加进了上一步的结果，这将导致更好的“雪崩效应”。SHA-1引入了一个新的变量TEMP，这个精巧的改变可抵抗文献[36]中对MD_5的攻击方法。

8.6 时　　戳

使用一个数字签名方案时，可能会遇到一些麻烦问题，诸如，一个签名者的签名算法

可能被一个敌手偷走，此时，对签名者在敌手偷走签名算法之前所作的签名的真实性会受到影响。也可能签名者签了名以后，他想否认签名，他就将签名算法公布于众，然后声称他对消息的签名是一个伪造签名。出现这些情况的主要原因是没有办法确定一个消息何时被签了名。解决这种问题的一个办法是使用时戳(timetamping)[38,39]。本节将描述一些时戳方法。

签名者B自己可产生一个可信的时戳。B首先获得目前公开可获得的某一信息，该信息在它发生之前不能预测。例如，这种信息可由前一天的某种股票交易额构成，将这个信息记为pub。

现在假定B想时戳他对消息 x 的签名。我们假定 h 是一个公开知道的Hash函数。B产生时戳的方法如下：

(1) B计算 $z=h(x)$；

(2) B计算 $z'=h(z \| \text{pub})$；

(3) B计算 $y=\text{Sig}_K(z')$；

(4) B在第二天的新闻报纸上公布 (z,pub,y)。

信息pub的存在意味着B在所谈到的日期之前不能产生 y。将 y 公布在第二天的新闻报纸上这一事实证明了B在所谈到的日期之后不能计算 y。所以B的签名 y 被限制在周期为一天的范围内。在这种方法中，B没有泄露 x，只有 z 被公开。必要时，B能证明 x 是他所签的消息。

如果存在一个可信的时戳服务中心TSS，那么B产生时戳的过程可改为：B先计算 $z=h(x)$ 和 $y=\text{Sig}_K(z)$，然后将 (z,y) 发送给TSS。TSS将级联日期 D 并对 (z,y,D) 进行签名。这种方法只能证明B在某一时间之前签了消息。如果B也想说明他在某一时间之后签了消息，他可以像上述一样插入某一公开知道的信息pub。

如果不希望无条件地相信TSS，那么可通过连续地联接被时戳的消息来增加安全性。在这样一种方案中，B将给TSS发送一个有序的三重组 $(z,y,ID(\text{B}))$。其中 z 是消息 x 的消息摘要，y 是B对 z 的签名，$ID(\text{B})$ 是B的识别信息。TSS将时戳一列这种形式的三重组，用 (z_n,y_n,ID_n) 表示将被TSS时戳的第 n 个三重组，用 t_n 表示第 n 次申请时戳的时间。TSS用下列方法来时戳第 n 个三重组：

(1) TSS计算 $L_n=(t_{n-1},ID_{n-1},z_{n-1},y_{n-1},h(L_{n-1}))$；

(2) TSS计算 $C_n=(n,t_n,z_n,y_n,ID_n,L_n)$；

(3) TSS计算 $S_n=\text{Sig}_{\text{TSS}}(h(C_n))$；

(4) TSS将 (C_n,S_n,ID_{n+1}) 发送给 ID_n。

L_n 是一个“联系信息”(linking information)，它将第 n 次申请和先前的申请连接在一起。如果发生争端，B能泄露他的消息 x_n，并能验证 y_n。TSS的签名 S_n 也能被验证。必要的话，ID_{n-1} 或 ID_{n+1} 能被要求产生他们的时戳，分别为 (C_{n-1},S_{n-1},ID_n) 和 $(C_{n+1},S_{n+1},ID_{n+2})$。在这些时戳中，TSS的签名都能被验证。当然，如果需要的话，这个过程可继续向前或向后进行。对时戳技术感兴趣的读者可参阅文献[38]和[39]。

8.7 注记和文献

杂凑(Hash)函数是密码学中的一种很有用的函数。关于杂凑函数，已有很多文献进

行了讨论，特别值得一提的是文献[1]，它是一本全面了解 Hash 算法的很好的参考书。另外，关于一些别的具体 Hash 算法可在文献[41]中找到线索。文献[42]是一篇很好的有关 Hash 算法的综述文献。

参 考 文 献

[1] Pieprzyk, J. And Sadeghiyan, B., Design of Hashing algorithms, Springer-Verlag, 1993.

[2] Rabin, M. O., Digitalized Signatures, In Demillo, P. A., Dobkin, D. P., Jones, A. K. And Lipton, R. J., editors, Foundations of Secure Computation, New York, 1978, Academic Press, pp. 155—166.

[3] Merkel, R. C., A Fast Software One-way Hash Function, Journal of Cryptology, Vol. 3, No. 1, 1990, pp. 43—58.

[4] Miyaguchi, S., Iwata, M. and Ohta, K., New 128-bit Hash Function, Proceedings of 4th International Joint Workshop on Computer and Communications, 1989, pp. 279—288.

[5] Biham, E. and Shamir, A., Differential Cryptanalysis of DES-like Cryptosystems, Advances in Cryptology-crypto'90, Springer-Verlag, 1991, pp. 2—21.

[6] Rivest, R., The MD_4Message Digest Algorithm, Advances in Cryptology-Crypto'90, Springer-Verlag, 1991, pp. 303—311.

[7] Rivest, R., The MD_5Message Digest Algorithm, RFC1321, Apr. 1992.

[8] Secure Hash Standard, National Bureau of Standards FIPS Publication 180, 1993.

[9] Schneier, B., Applied Cryptography-protocols, Algorithms, and Source Code in C, John Wiley & Sons, Inc., 1993.

[10] Menezed, A. J., Van Oorschot, P. C. and Vanstone, S. A., Handbook of Applied Cryptography, IEEE Press, 1996.

[11] Merkle, R. C., A Certified digital Signature, Advances in Cryptology-Crypto'89, Srpinger-Verlag, 1990, pp. 218—218.

[12] Carter, J. L. and Wegman, M. N., Universal Classes of Hash Functions, Journal of Computer and System Sciences, 1979, No. 18, pp. 143—154.

[13] Stinson, D. R., Cryptography——Theory and Practice, CRC Press, 1995.

[14] ANSI X9. 9 (Revised), American National Standard for Financial Institution Message Authentication, American Bankers Association, 1986.

[15] ISO/IEC 9797, Data Cryptographic Techniques Data Integrity Mechanism Using a Cryptographic Check Function Employing a Block Cipher Algorithm, International Organization for Standardization, 1989.

[16] Preneel, B., Cryptographic Hash Functions, ETT, Vol. 5, No. 4, pp. 17/431—31/455.

[17] Damgard, I. B., A Design Principle for Hash Functions, Advances in Cryptology—— Crypto'89, Springer-Verlag, 1990, pp. 416—427.

[18] Merkel, R. C., One Way Hash Functions and DES, Advances in Cryptology—— Crypto'89, Springer-Verlag, 1990, pp. 428—446.

[19] Ohta, K. and Koyama, K., Meet-in-the-middle Attack on Digital Signature Schemes, Advances in Cryptology—— auscrypt'90, Springer-Verlag, 1991, pp. 110—121.

[20] Nishimura, K. and Sibuya, M., Probability to Meet in the Middle, Journal of Cryptology, No. 2, 1990, pp. 13—22.

[21] Winternitz, R. S., Producing a One-way Hash Function from DES, Advances in Cryptology-Crypto'83, Plenum Publishing Corporation, 1984, pp. 203—207.

[22] Davies, D. W. and Price, W. L., The Application of digital Signatures Based on Public Key Cryptosystems, In Proceedings of the Fifth International Conference on Computer Communication, 1980, pp. 525—530.

[23] Davies, D. W. and Price, W. L., Digital Signature—— an Update, In Proceedings of the Seventh International Conference on Computer Communication, 1984, pp. 845—849.

[24] Coppersmith, D., Another Birthday Attack, Advances in Cryptology-Crypto'85, Springer-Verlag, 1986, pp. 14—17.

[25] Girault, M., Cohen, R. and Campana, M., A Generalized Birthday Attack, Advances in Cryptology——

Eurocrypt'88, Springer-Verlag, 1989, pp. 129—156.

[26] Mitchell, C. and Walker, M., Solutions to the Multidestination Secure Electronic Mail Problem, Computers & security, No. 7, 1988, pp. 483—488.

[27] Mitchell, C., Multi-destination Secure Electronic Mail, The Computer Journal, No. 32, 1989, pp. 13—15.

[28] Biham, E. and Shamir, A., Differential Cryptanalysis of the Data Encryption Standard, Springer-Verlag, 1993.

[29] Berson, T. A., Differential Cryptanalysis Mod 2^{32} with Applications to MD_5, Advances in Cryptology—— Eurocrypt'92, Springer-Verlag, 1993, pp. 67—76.

[30] Miyaguchi, s., Ohta, K. and Iwata, M., Confirmation That Some Hash Functions Are Not Collision Free, Advances in Cryptology—— Eurocrypt'90, Springer-Verlag, 1991, pp. 293—308.

[31] Gilbert, E. N., Macwilliams, F. J. and Sloane, N. J., Codes which Detect Deception, Bell System Technical Journal, 1974, 53, pp. 405—425.

[32] Simmons, G. J., Authentication Theory/coding Theory, Advances in Cryptology Crypto'84, Springer-Verlag, 1985, pp. 411—431.

[33] Chaum, D., Van Heijst, E. and Pfitzmann, B., Cryptographically Strong Undeniable Signatures, Uncoditionally Secure for the Signer, Advances in Cryptology——Crypto'91, Springer-Verlag, 1992, pp. 470—484.

[34] Den Boer, B. and Bosselaers, A., An Attack on the Last Two Rounds of MD_4, Advances in Cryptology—— Crypto'91, Springer-Verlag, 1992, pp. 194—203.

[35] Berson, T., Differential Cryptanalysis Mod 2^{32} with Applications to MD_5, Advances in Cryptology—— Eurocrypt'92, Springer-Verlag, 1993, pp. 71—80.

[36] Den Boer, B. and Bosselaers, A., Collisions for the Compress Function of MD_5, Advances in Cryptology—— Eurocrypt'93, Springer-Verlag, 1994, pp. 293—304.

[37] Zheng, Y., Pieprzyk, J. and Seberry, J., Haval a One-way Hashing Algorithm with Variable Length of Output, Advances in Cryptolog-Auscrypt'92, Springer-Verlag, 1993, pp. 83—104.

[38] Heber, S. and Stornetta, W. S., How to Timestamp a Digital Document, Journal of Cryptology, 3 (1991), pp. 99—111.

[39] Bayer, D., Haber, S. and Stornetta, W. S., Improving the Efficiency and Reliability of Digital Time-stamping, In Sequences Ⅱ, Methods in Communication, Security, and Computer Science, Springer-Verlag, 1993, pp. 309—334.

[40] Mitchell, C. J. Piper, F. and Wild, P., Digital Signatures, In Contemporary Cryptology, the Science of Information Integrity, IEEE Press, 1992, pp. 325—378.

[41] Schneier, B., Applied Cryptography-protocols, Algorithms, and Source Code in c. John Wiley & Sons, Inc., 1993.

[42] Preneel, B., Govaerts, R., Vandewalle, J., Information Authentication: Hash Functions and Digital Signautures, Computer Security and Industrial Cryptography, Srpinger-Verlag, 1993, pp. 87—131.

第9章 识别协议

当一个用户A登录进入计算机(或自动出纳机(ATM)、电话银行系统等)时,计算机如何知道他是A呢?计算机又如何知道他不是其他人伪造他的身份呢?传统的办法是用通行字来解决这个问题。A先输入他的通行字,然后计算机确认他的正确性。A和计算机都知道这个秘密通行字,A每次登录时,计算机都要求A输入通行字。人们注意到计算机无需知道通行字,它有能力区别有效通行字和无效通行字。这种办法是通过用单向函数来实现的[1,2,3]。计算机存储通行字的单向函数值而不是存储通行字。其认证过程是:

(1) A将他的通行字传送给计算机;

(2) 计算机完成通行字的单向函数值的计算;

(3) 计算机把单向函数值和机器存储的值比较。

由于计算机不再存储每个人的有效通行字表,某些人侵入计算机也无法从通行字的单向函数值表中获得通行字。但这种保护也是很脆弱的,它抵抗不住字典式攻击[4,5]。所谓字典式攻击是指用单向函数对常用的通行字运算,将运算结果存储起来,并与计算机中存储的单向函数表对照,找出匹配的通行字。为了抵抗字典式攻击,人们建议采用salt方法,即将一个随机串salt与通行字连接在一起,再用单向函数对其运算,然后将salt值和单向函数值存入数据库中。这样做仍有严重的安全问题。首先,当A将他的通行字输入系统时,能够访问他的数据通道的任何人都可以读取他的通行字。其次在系统对通行字加密前,能访问计算机系统的存储器的任何人都可以看到通行字。

目前在实际中所使用的一些别的技术诸如ID卡(身份证)、信用卡,个人识别号(PIN)等都存在着一些安全问题。这并不是说我们不能解决识别技术的安全问题,利用密码技术我们就可以设计出安全的识别协议(方案)。

一个安全的识别协议至少应满足以下两个条件:

(1) 证明者A能向验证者B证明他的确是A;

(2) 在证明者A向验证者B证明他的身份后,验证者B没有获得任何有用的信息,B不能模仿A向第三方证明他是A。

这两个条件是说证明者A能向验证者B电子地证明他的身份,而又没有向B泄露他的识别信息。目前已设计出了许多满足这两个条件的识别协议。从实用角度讲,我们最关心的是设计简单的而且能在一个Smart卡上实现的安全识别方案。Smart卡本质上是一个装有一个能完成算术运算的芯片的信用卡(credit card)。因此,需求的计算量和储存量都应保持尽可能的小。用这样的一个卡代替目前使用的ATM卡将是更安全的。但是,因为只用卡证明他的身份,我们对一个丢失的卡没有额外的保护,所以我们必须用额外的安全性来监控卡。为了确保只有卡的真正拥有者才能触发识别协议,拥有一个个人识别号(PIN)仍然是必要的。

在本章中,我们将介绍一些比较流行的、有代表性的识别协议。在这里我们先给一个基于任何私钥密码体制诸如DES的简单方案以饱眼福。这种协议称为口令-应答(chal-

lenge-and-response)协议。在这种协议中,我们假定证明者A向验证者B识别自己,A和B共享一个共同的秘密密钥k,k确定一个加密变换E_k。该协议的执行过程为:

(1) B随机选择一个口令x(x是一个64比特长的随机串),并将x发送给A;

(2) A计算$y=E_k(x)$并将y发送给B;

(3) B计算$y'=E_k(x)$并验证是否有$y=y'$。

上述协议的安全性是建立在A和B相互信任的基础之上。然而,在大量的应用场合,往往他们不一定相互信任甚至可能是敌对的,所以更有用的方案应该是不需要共享秘密,这一观点将贯彻到本章所介绍的其余方案中。但有一点是共同的,识别方案本质上都是口令-应答协议。

9.1 Feige-Fiat-Shamir识别协议和识别协议向签名方案的转化

9.1.1 Feige-Fiat-Shamir识别协议

Fiat和Shamir在1986年基于零知识证明的思想提出了一种新型的识别协议[6],后经Feige、Fiat和Shamir的改进成为身份的零知识证明[7]。这是最著名的身份的零知识证明。关于Feige-Fiat-Shamir身份识别协议已在本书第2章中作了详细描述。1986年7月9日,Feige-Fiat-Shamir向美国专利局提交他们的身份识别协议申请专利[8]。由于这种思想在美国军方的潜在应用,申请由军方审阅,引出了众所周知的"零知识证明与国防部"事件[9]。可见,这一身份识别协议及其思想非同小可,应予以重视。这也正是作者重申已在本书第2章中作过介绍的FFS识别方案的动机。

将身份信息嵌入协议中是可能的。假定I是一个代表A身份的二进制串,I包括名字、性别、民族、生日、文化程度、爱好、职业、住址等信息。用一个单向Hash函数计算$H(I,j)$,这里j是级联在I后面的一个小随机数。找出一系列使得$H(I,j)$为一模n的平方剩余的j值。不妨设$v_j=H^2(I,j)\bmod n$,$(j=1,2,\ldots,k)$。A公开I和这一串j值。在执行协议的第一步以前,他将I和这一串j值发送给B,B从$H(I,j)$产生$v_1,v_2,\ldots,v_k$。当B和A成功地执行完协议之后,B相信I是A的身份。

对非理想的Hash函数,在I后级联一个长的随机串R使之随机化是可取的。这个串由可信中心选取,随I发送给B[6]。

在典型实现中,k应该在1到18之间。较大的k值能减少时间和通信复杂度,这是通过减少(协议)轮数来实现的。n应至少为512比特长。如果所有的用户选取他们自己的n,并在一个公开文件中公布,他们可不必需要可信中心。然而,这种做法使方案变得使用起来不方便。

9.1.2 识别协议向签名方案的转化

有一个标准的方法可将一个识别协议转化成一个签名方案。基本的观点是用一个公开的Hash函数来代替验证者B。可见,每一个识别协议都可派生出一个数字签名方案,因此,在以下的讨论中每介绍一个识别协议就相应地介绍一个签名方案。现在,我们用Feige-Fiat-Shamir识别协议为例来图示如何将一个识别协议转化为一个签名方案,该方

案将称为 Fiat-Shamir 签名方案[6]。

可信中心随机选取一个模数 n，n 为两个大素数之积。实际中，n 至少为 512 比特长，可能接近 1024 比特。这个 n 值可以在一组签名者之间共享。可信中心选取 k 个不同的数 $v_1, v_2, \ldots, v_k$，这里 v_i 为一个模 n 的平方剩余。计算 $s_i = 1/v_i^{1/2} \bmod n, 1 \leqslant i \leqslant k$。签名者 A 的公开密钥为 $v_1, v_2, \ldots, v_k$，秘密密钥为 $s_1, s_2, \ldots, s_k$。设 A 使用的 Hash 函数为 $H(.), H(.)$ 是公开知道的。

A 对消息 m 的签名过程如下：

(1) A 在[0,n)中随机选择 t 个整数并计算 $x_i = r_i^2 \bmod n$，$1 \leqslant i \leqslant t$；

(2) A 计算 $H(m, x_1, x_2, \ldots, x_t)$，用 $e_{ij}(1 \leqslant i \leqslant t,\ 1 \leqslant j \leqslant k)$表示 $H(m, x_1, x_2, \ldots, x_t)$ 的前 kt 比特；

(3) A 计算 $y_i = r_i \prod_{e_{ij}=1} s_i \bmod n$ $(1 \leqslant i \leqslant t)$并将 $m, y_i(1 \leqslant i \leqslant t)$和 $e_{ij}(1 \leqslant i \leqslant t,\ 1 \leqslant j \leqslant k)$发给接收者 B。

B 验证 A 对 m 的签名的过程为：

(1) B 计算 $z_i = y_i^2 \prod_{e_{ij}=1} v_j \bmod n, 1 \leqslant i \leqslant t$；

(2) B 验证 $H(m, z_1, z_2, \ldots, z_k)$的前 kt 个比特是否与 $e_{ij}(1 \leqslant i \leqslant t,\ 1 \leqslant j \leqslant k)$一致。

Fiat-Shamir 数字签名与 RSA 签名相比，主要的优点是运算速度快，只需要 RSA 的模乘法次数的 1%～4%。他们推荐 kt 应从 20～72，并建议使用 $k=9, t=8$。关于其安全性分析详见文献[6]。

文献[10]改进了 Fiat-Shamir 识别和签名方案。他们将 $v_1, v_2, \ldots, v_k$ 选择为前 k 个素数。因而 $v_1=2, v_2=3, v_3=5$ 等，这是公开密钥。秘密密钥 $s_1, s_2, \ldots, s_k$ 是随机平方根，由 $s_i = \sqrt{v_i^{-1}} \bmod n$ 给出。这种改进的方案中，每个人有不同的 n 值。这个改进带来的好处是使得验证签名变得容易，而产生签名所需的时间和这些签名的安全性不受影响。基于 Fiat-Shamir 识别和签名方案的其它改进，感兴趣的读者请参阅文献[11,12]。

9.2 Schnorr 识别协议

Schnorr 协议[13]融合了 ElGamal 协议、Fiat-Shamir 协议和 Chaum-Evertse-Van de Graff 交互式协议[14]等协议的思想，其安全性建立在计算离散对数问题的困难性之上。该协议是最有吸引力的实用识别协议之一，在许多国家都申请了专利。

Schnorr 协议需要一个可信中心，记为 TA。TA 将为协议选择下列一些参数：

(1) p 是一个大素数($p \geqslant 2^{512}$)，在 Z_p^* 上计算离散对数是难处理的；

(2) q 是一个大素数($q \geqslant 2^{140}$)，并且 $q \mid (p-1)$；

(3) $\alpha \in Z_p^*$ 有阶 q(诸如可取 $\alpha = g^{(p-1)/q}$，g 为 Z_p 的本原元)；

(4) 一个安全参数 t，$q > 2^t$，对大多数应用来说，取 $t=40$ 将提供足够的安全性，为了更高的安全性，Schnorr 建议[13]使用 $t=72$；

(5) TA 选择一个安全的签名方案，秘密签名算法为 Sig_{T_A}，公开验证算法为 Ver_{T_A}；

(6) 选定一个安全的 Hash 函数。像通常一样，所有的信息在签名之前先进行杂凑，为

了协议的可读性，我们在描述协议时将略去杂凑这一步。

参数 p、q、α、Ver_{T_A}和 Hash 函数都是公开的。

TA 将给每个用户都要颁布一个证书。当 A 想从 TA 那里获得一个证书时，A 和 TA 执行下列协议：

(1) TA 建立 A 的身份，TA 形成一个串 ID_A，ID_A 包含 A 的足够多的信息，诸如姓名

能在多项式时间内计算出 a。

证明：在 2^t 个可能的口令 r 中，大约有 $2^t\times\varepsilon$ 个口令，O 能计算一个 y 值使 B 在识别协议中的第(6)步接收。因为 $\varepsilon\geq 1/2^{t-1}$，所以 $2^t\times\varepsilon\geq 2$，故 O 能计算 y_1、y_2、r_1 和 r_2 使得 $r_1\neq r_2$ 且 $\gamma\equiv\alpha^{y_1}v^{r_1}\equiv\alpha^{y_2}v^{r_2}\bmod p$，即 $\alpha^{y_1-y_2}\equiv v^{r_2-r_1}\bmod p$。因为 $v=\alpha^{-a}$，所以 $y_1-y_2\equiv a(r_1-r_2)\bmod q$。又因为 $0<|r_2-r_1|<2^t$ 且 $q>2^t$ 是素数，所以 $\gcd(r_2-r_1,q)=1$，故 O 能计算 $a=(y_1-y_2)(r_1-r_2)^{-1}\bmod q$。

上述定理表明，能以一个不可忽略的概率成功地执行识别协议的任何人都一定知道(即能在多项式时间内计算出)A 的秘密指数 a。换句话说，如果假定 O 不知道 A 的秘密指数 a，那么他成功地执行识别协议的概率可忽略(很小)。这说明 Schnorr 识别协议满足合理性(与第 2 章中协议的合理性的定义一致)。

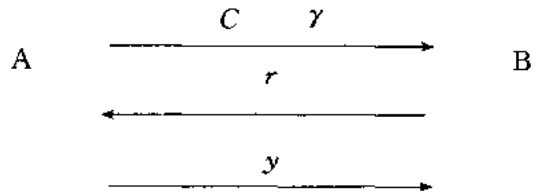
一个满足完全性和合理性的协议并不能保证协议是安全的。例如，A 可以通过简单地泄露他的指数 a 的值来向 O 证明他的身份，该协议仍然是完全的和合理的。然而，这个协议是完全不安全的，因为 O 能模仿 A。在密码学中，我们希望一个协议能够在 A 向 O 证明他的身份时没有泄露 a 的任何信息。这就是我们在第 2 章中介绍的零知识思想。目前为止，仍然没有证明 Schnorr 协议是安全的。不过，它的一个修改即下节将要介绍的 Okamoto 协议可证明是安全的。

Schnorr 协议从计算量和需要交换的信息量两方面来看都是很快的和有效的。它也极小化了由 A 所完成的计算量。这是因为在许多实际应用中，A 的计算将由一个低计算能力的 Smart 卡来完成，而 B 的计算将由一个具有较强计算能力的计算机来完成。

为了说明问题，我们假定 ID(A)和 v 的长度均为 512 比特。如果 TA 使用 DSS 签名，那么 s 的长度为 320 比特。此时，证书 C(A)的总长度为 1344 比特。

让我们考虑一下 A 的计算：第(1)步需要完成一个模指数运算；第(5)步需要一个模加法运算和一个模乘法运算。只有模指数运算是复杂的，需要的时间较长，但这个可以离线预计算。由 A 完成的在线运算是很适中的。

我们用下图来表示和通信的信息：



A 在第(2)步给 B 发送了 1344+512=1856 比特长的信息；B 在第(4)步给 A 发送了 40 比特长的信息；而 A 在第(6)步给 B 发送了 140 比特长的信息。所以通信量也是很适中的。

Schnorr 签名方案[13]

设 p 是一个大素数，在 Z_p 上的离散对数问题是难处理的，q 也是一个大素数且 $q|(p-1)$。设 $\alpha\in Z_p^*$ 是一个阶为 q 的元素。H 是一个 Hash 函数。$\boldsymbol{P}=Z_p^*$，$\boldsymbol{A}=Z_p^*\times Z_q$，$\boldsymbol{K}=\{(p,q,\alpha,a,v)\mid\equiv\alpha^{-a}\bmod p\}$。值 p、q、α、v 和 H 是公开的，a 是保密的。对 $K=(p,q,\alpha,a,V)$ 和一个(秘密的)随机数 $k\in Z_q^*$，定义 $\mathrm{Sig}_K(x,k)=(\gamma,y)$，其中 $\gamma=\alpha^k\bmod p$，$y=k+aH$

$(x,\gamma)\bmod q$。对 $x,\gamma\in Z_p^*$ 和 $y\in Z_q$，定义 $\mathrm{Ver}(x,\gamma,y)=$真$\Leftrightarrow\gamma\equiv\alpha^y v^{H(x,\gamma)}\bmod p$。

文献[15]中对 Schnorr 方案的预处理过程进行了分析。文献[16]对 Schnorr 方案作了另一种改进，增强了其安全性。

9.3 Okamoto 识别协议

Okamoto 识别协议[17]是 Schnorr 协议的一种改进，这种改进使得在假定在 Z_p 上计算一个特定的离散对数是难处理的情况下，可证明其安全性。但仅就从速度和有效性来讲，Schnorr 协议比 Okamoto 协议更实用。

为了建立方案，TA 像在 Schnorr 方案中一样，选择两个大素数 p 和 q。TA 也在 Z_p 中选择两个阶为 q 的元素 α_1,α_2。对系统的所有参加者包括 A，TA 保密 $c=\log_{\alpha_1}\alpha_2$。我们假定任何人(即使 A 和 O 合伙)计算值 c 是不可行的。像在 Schnorr 方案中一样，TA 选择一个签名方案和一个 Hash 函数。

TA 向 A 颁布一个证书的协议为：

(1) TA 建立 A 的身份并颁布一个识别串 ID(A)；

(2) A 秘密地选择两个随机指数 a_1、a_2，$0\leqslant a_1,a_2\leqslant q-1$，计算 $v=\alpha_1^{-a_1}\alpha_2^{-a_2}\bmod p$ 并将 v 发送给 TA。

(3) TA 对(ID,v)签名，$s=\mathrm{Sig}_{T_A}(\mathrm{ID},v)$。TA 将证书 $C(\mathrm{A})=(\mathrm{ID}(\mathrm{A}),v,s)$发送给 A。

Okamoto 识别协议为：

(1) A 随机选择两个数 k_1、k_2，$0\leqslant k_1,k_2\leqslant q-1$，并计算 $\gamma=\alpha_1^{k_1}\alpha_2^{k_2}\bmod p$；

(2) A 将他的证书 $C(\mathrm{A})=(\mathrm{ID}(\mathrm{A}),v,s)$和 γ 发送给 B；

(3) B 通过检测 $\mathrm{Ver}_{T_A}(\mathrm{ID}(\mathrm{A}),v,s)=$真来验证 TA 的签名；

(4) B 随机选择一个数 r，$1\leqslant r\leqslant 2^t$，并将 r 发送给 A；

(5) A 计算 $y_1=(k_1+a_1r)\bmod q$，$y_2=(k_2+a_2r)\bmod q$ 并将 y_1、y_2 发送给 B；

(6) B 验证 $\gamma\equiv\alpha_1^{y_1}\alpha_2^{y_2}v^r\bmod p$。

易知，如果 A 遵循协议，那么 B 将接收 A 的身份证明。这表明 Okamoto 协议满足完全性。类似于 Schnorr 方案的合理性的证明，我们可证明如下定理：

定理 9.3.1 假定 O 知道一个值 γ，他能用此值以概率 $\epsilon\geq 1/2^{t-1}$成功地模仿 A，则 O 能在多项式时间内计算出 b_1 和 b_2 使得 $v\equiv\alpha_1^{-b_1}\alpha_2^{-b_2}\bmod p$。

证明：在 2^t 个可能的口令 r 中，大约有 $2^t\times\epsilon$ 个口令，O 能计算一对 y_1 和 y_2 使 B 在识别协议中的第(6)步接收。因为 $\epsilon\geq 1/2^{t-1}$，所以 $2^t\times\epsilon\geq 2$，故 O 能计算 y_1、y_2、z_1、z_2、r 和 s 使得 $r\neq s$ 且 $\gamma\equiv\alpha_1^{y_1}\alpha_2^{y_2}v^r\equiv\alpha_1^{z_1}\alpha_2^{z_2}v^s\bmod p$。令 $b_1=(y_1-z_1)(r-s)^{-1}\bmod q$，$b_2=(y_2-z_2)(r-s)^{-1}\bmod q$，易验证 $v\equiv\alpha_1^{-b_1}\alpha_2^{-b_2}\bmod p$。

Okamoto 协议和 Schnorr 协议的主要差别在于：在假定计算离散对数 $\log_{\alpha_1}\alpha_2$ 是不可行的情况下，我们能证明 Okamoto 方案是安全的。其安全性证明的基本思路是：A 通过执行协议多项式次向 O 识别自己。假定 O 能获得 A 的秘密指数 a_1 和 a_2 的某些信息，我们将证明 A 和 O 一起能以很高的概率在多项式时间内计算离散对数 c，这与我们的假设矛盾。这样我们就证明了 O 通过参加协议一定不能获得关于 A 的指数的任何信息。现在我们来详细证明 Okamoto 协议的安全性。

首先我们说明 A 和 O 怎样一起来计算 c 的值。

定理 9.3.2 假定 O 知道一个值 γ，他能用此值以概率 $\varepsilon \geq 1/2^{t-1}$ 成功地模仿 A，则 A 和 O 合伙能以概率 $1-1/q$ 在多项式时间内计算 $\log_{a_1} a_2$。

证明：由定理 9.3.1 可知，O 能确定值 b_1 和 b_2 使得 $v \equiv a_1^{-b_1} a_2^{-b_2} \bmod p$。

现在假定 A 将值 a_1 和 a_2 泄露给 O。当然有 $v = a_1^{-a_1} a_2^{-a_2} \bmod p$，所以。

$$a_1^{a_1-b_1} \equiv a_2^{b_2-a_2} \bmod p$$

现在分两种情况讨论：

情况 1：$(a_1, a_2) \neq (b_1, b_2)$。此时 $a_1 \neq b_1$ 且 $a_2 \neq b_2$，从而

$$(a_1 - b_1)^{-1} \bmod q \text{ 和 } (b_2 - a_2)^{-1} \bmod q$$

都存在，离散对数 $c = \log_{a_1} a_2 = (a_1 - b_1)(b_2 - a_2)^{-1} \bmod q$ 可在多项式时间内计算出。

情况 2：$(a_1, a_2) = (b_1, b_2)$。此时我们不能利用上面的公式计算 c 的值。然而，我们能说明 $(a_1, a_2) = (b_1, b_2)$ 只能以很小的概率 $1/q$ 发生，所以由 A 和 O 合伙计算 c 几乎必定成功。

定义

$$\mathbf{A} = \{(a_1', a_2') \in Z_q \times Z_q \mid a_1^{-a_1'} a_2^{-a_2'} \equiv a_1^{-a_1} a_2^{-a_2} \bmod p\}$$

即 **A** 由所有可能的 A 的秘密指数有序对构成。**A** 可表示为

$$\mathbf{A} = \{(a_1 - c\theta, a_2 + \theta) \mid \theta \in Z_q\}, c = \log_{a_1} a_2$$

这样 $|\boldsymbol{A}| = q$。由 O 合伙计算的有序对 (b_1, b_2) 一定在集合 **A** 之中。我们将说明有序对 (b_1, b_2) 的值独立于由 A 秘密选择的有序对 (a_1, a_2) 的值。如果这一点得到证明，那么因为 (a_1, a_2) 是当初由 A 随机选择的，所以 $(a_1, a_2) = (b_1, b_2)$ 的概率一定为 $1/q$。我们需要说明的是我们所说的 (b_1, b_2) 独立于 (a_1, a_2) 意指 A 选择的对 (a_1, a_2) 是 **A** 中的 q 个可能的有序对中之一，并且当 A 向 O 识别他的身份时，没有泄露给 O 关于正确有序对即 (a_1, a_2) 的任何信息。也就是说，O 知道在 **A** 中有一个有序对构成了 A 的指数，但他没有办法区别开来究竟 A 使用了这 q 个有序对中的哪一个。

让我们看一看在识别协议中交换的信息。

在协议的每一次执行中，A 选择一个 γ，O 选择一个 r，A 泄露使得 $\gamma \equiv \alpha_1^{y_1} \alpha_2^{y_2} v^r \bmod p$ 的 y_1 和 y_2。其中 $y_1 = (k_1 + a_1 r) \bmod q$，$y_2 = (k_2 + a_2 r) \bmod q$，$\gamma \equiv \alpha_1^{k_1} \alpha_2^{k_2} \bmod p$。$k_1$ 和 k_2，a_1 和 a_2 都没有被泄露。

在协议的每一次执行中产生的特定的四重组 (γ, r, y_1, y_2) 似乎依赖于 A 的有序对 (a_1, a_2)，因为 y_1 和 y_2 分别是 a_1 和 a_2 的函数。但是我们将证明每一个这样的四重组能等可能地从 **A** 中的任何一对别的有序对 (a_1', a_2') 产生出。为了证明这一点，假定 $(a_1', a_2') \in \mathbf{A}$，即 $a_1' = a_1 - c\theta$，$a_2' = a_2 + \theta$，$0 \leq \theta \leq q-1$。y_1 和 y_2 可表示为

$$y_1 = k_1 + a_1 r = (k_1 + rc\theta) + a_1' r$$

$$y_2 = k_2 + a_2 r = (k_2 - r\theta) + a_2' r$$

这里的所有的算术运算都是 Z_q 中的运算。由 y_1 和 y_2 的表示式可知，四重组 (γ, r, y_1, y_2) 也可使用有序对 (a_1', a_2') 及随机数 $k_1' = k_1 + rc\theta$ 和 $k_2' = k_2 - r\theta$ 产生。

我们已经注意到，k_1 和 k_2 的值没有被 A 公开，所以不管 A 使用 **A** 中哪一个有序对，四重组 (γ, r, y_1, y_2) 都没有给出 A 实际上使用的秘密指数的任何信息。

现在我们将上述证明 Okamoto 协议的安全性的要点作一归纳：

在 Okamoto 协议中，A 选择了两个秘密指数而不是一个，在 **A** 中总共有 q 对等价于 A 的对(a_1,a_2)。导致最终矛盾的事实是 **A** 中两个不同对的知识将提供计算离散对数 c 的一个有效方法。当然，A 知道 **A** 中的一个对，并且我们证明如果 O 能模仿 A，那么 O 能计算 **A** 中的一个对，这个对与 A 的对不同的概率很大。这样 A 和 O 合伙就能找到 **A** 中的两个不同的对来计算 c，从而导出了一个矛盾。

Okamoto 签名算法[17]

系统的参数为 p、q、α_1、α_2 和 t，$q|(p-1)$，p 和 q 为大素数，p 至少为 512 比特长，q 至少为 140 比特长，t 是一个安全参数，t 至少为 20，α_1 和 α_2 都是 Z_p 中阶为 q 的元素。签名者 A 的公开密钥为 $v=\alpha_1^{-a_1}\alpha_2^{-a_2}\bmod p$，秘密密钥是一对参数 a_1、a_2，$a_1,a_2\in Z_q$。H 是一个单向函数。

当 A 对消息 m 签名时，首先随机选择两个数 $r_1,r_2\in Z_q$，然后计算：

$$x=\alpha_1^{r_1}\alpha_2^{r_2}\bmod p$$

$$e=H(x,m)$$

$$y_1=(r_1+es_1)\bmod q$$

$$y_2=(r_2+es_2)\bmod q$$

A 对消息 m 的签名是三重组(e,y_1,y_2)。

当接收者 B 收到签名(e,y_1,y_2)时，B 计算 $x=\alpha_1^{y_1}\alpha_2^{y_2}v^e\bmod p$，并验证 $e=H(x,m)$。

9.4 Guillou-Quisquater 识别协议

Guillou-Quisquater 识别协议[18]的安全性是基于 RSA 体制的安全性。可证明 Guillou-Quisquater 识别协议满足完全性和合理性，但即使在假定 RSA 体制安全的情况下，也没有被证明该协议是安全的。

协议的建立过程如下：TA 选择两个大素数 p 和 q，形成 $n=pq$。保密 p 和 q，公开 n。TA 选择一个大素数 b(b 用作一个安全参数)和一个公开的 RSA 加密指数。一般假定 b 是一个 40 比特长的素数。TA 选择一个签名方案和 Hash 函数。

TA 向证明者 A 颁布一个证书的协议如下：

(1) TA 建立 A 的身份并颁布一个识别串 ID(A)；

(2) A 秘密地选择一个整数 u，$0\leqslant u\leqslant n-1$。A 计算 $v=(u^{-1})^b\bmod n$ 并将 v 发送给 TA；

(3) TA 对(ID,v)签名，$s=\mathrm{Sig}_{T_A}(\mathrm{ID},v)$。并将证书 $C(\mathrm{A})=(\mathrm{ID}(\mathrm{A}),v,s)$发送给 A。

证明者 A 向验证者 B 证明他的身份的协议即 Guillou-Quisquater 识别协议为：

(1) A 选择一个随机数 k，$0\leqslant k\leqslant n-1$，并计算 $\gamma=k^b\bmod n$；

(2) A 把他的证书 $C(\mathrm{A})=(\mathrm{ID}(\mathrm{A}),v,s)$和 γ 发送给 B；

(3) B 通过检测 $\mathrm{Ver}_{T_A}(\mathrm{ID}(\mathrm{A}),v,s)=$真来验证 TA 的签名；

(4) B 选择一个随机数 r，$0\leqslant r\leqslant b-1$，并将 r 发送给 A；

(5) A 计算 $y=ku^r\bmod n$ 并将 y 发送给 B；

(6) B 验证 $\gamma\equiv v^r y^b\bmod n$。

直接可说明 Guillou-Quisquater 识别协议满足完全性。现在，我们来考虑该协议的合理性。假定从 v 计算 u 是不可行的，那么我们可证明该协议满足合理性。因为 v 是由 u 通过 RSA 加密形成的，所以做这种假定似乎是合理的。

定理 9.4.1 假定 O 知道一个值 γ，他用该值成功地模仿 A 的概率 $\varepsilon>1/b$，则 O 能在多项式时间内计算出 u。

证明：对 γ，O 能计算值 y_1、y_2、r_1、r_2 使得 $r_1\neq r_2$ 且 $\gamma\equiv v^{r_1}y_1^b\equiv v^{r_2}y_2^b \bmod n$。不失一般性，假定 $r_1>r_2$，则 $v^{r_1-r_2}\equiv(y_2/y_1)^b \bmod n$。因为 $0<r_1-r_2<b$ 且 b 是素数，所以 $t=(r_1-r_2)^{-1} \bmod b$ 存在。O 利用 Euclidean 算法可在多项式时间内计算出 t。因此，$v^{(r_1-r_2)t}\equiv(y_2/y_1)^{bt} \bmod n$。设 $(r_1-r_2)t=lb+1$，l 为某一正整数，即 $l=((r_1-r_2)t-1)/b$。这样 $v^{lb+1}\equiv(y_2/y_1)^{bt} \bmod n$，即 $v\equiv(y_2/y_1)^{bt}(v^{-1})^{lb} \bmod n$，从而 $u^{-1}\equiv v^{b^{-1}\bmod\varphi(n)}\equiv(y_2/y_1)^t(v^{-1})^l \bmod n$，即 $u\equiv(y_1/y_2)^t v^l \bmod n$。O 使用这个公式可在多项式时间内计算出 u。

Guillou-Quisquater 签名方案[18]

签名者 A 选择两个大素数 p 和 q，形成 $n=pq$。A 选择一个大素数 b 和一个单向 Hash 函数 H。A 秘密选择一个整数 u，$\gcd(u,n)=1$，将 u 作为他的秘密密钥。A 的公开密钥为 $v=(u^{-1})^b \bmod n$。公开 n,b,v,H，保密 p,q 和 u。

A 对消息 m 的签名过程为：

(1) 随机选择一个整数 k，$0\leq k\leq n-1$，$r=k^b \bmod n$；

(2) 计算 $e=H(m,r)$；

(3) 计算 $y=ku^e \bmod n$，A 对消息 m 的签名是对 (e,y)。

接收者 B 验证签名的过程为：

(1)获得 A 的公钥 n,b,v；

(2)计算 $r'=y^b v^e \bmod n$ 和 $e'=H(m,r')$；

(3)验证是否有 $e=e'$，如果 $e=e'$，则 B 接收 A 的签名，否则，拒绝。

9.5 基于身份的识别方案

9.5.1 Shamir 的基于身份的密码方案的基本思想

Shamir 在 1984 年提出了一类新型的密码方案[20]，该类方案能使网上的任何一对用户无需交换秘密密钥或公开密钥、无需保存密钥簿、无需使用第三方服务可进行安全的通信和相互验证签名。在这类方案中，假定了存在一个可信的密钥产生中心，该中心的主要作用是给每一个第一次入网的用户颁发一个个人化的 Smart 卡。嵌入在这个卡中的信息能使用户签名和加密他所发送的消息，并且能使用户用一个完全独立的方式解密和验证他所收到的消息。

Shamir 的基于身份的密码方案仍然是一类公钥密码方案，但它不是去直接生成一对随机的公开密钥和秘密密钥，而是由用户选择他的名字和网络地址来作为公开密钥。当用户入网时，密钥产生中心首先对用户进行识别。如果接纳用户，密钥产生中心就为该用户以 Smart 卡的形式颁布一个秘密密钥。该卡包含一个微处理器、一个 I/O 端口、一个 RAM、一个带秘密密钥的 ROM 以及加密/解密消息和产生/验证签名的程序。于是，当一

个网上用户想与另一个用户通信时，他只需知道对方的姓名和地址就行了。这种方案与我们目前使用的邮政系统十分类似。

当用户 A 想给用户 B 发送一个消息 m 时，他就用自己的 Smart 卡中的秘密密钥对消息 m 签名，用 B 的名字和网络地址来加密签名和消息 m，然后将密文连同 A 的名字和地址发送给 B。当 B 收到消息时，他使用他的 Smart 卡中的秘密密钥对消息进行解密，最后用发送者的名字和网络地址作为验证密钥验证签名。

基于身份的密码方案的安全性主要依赖于以下几个方面：

(1) 所使用的密码变换(诸如加密变换、签名变换等)的安全性；

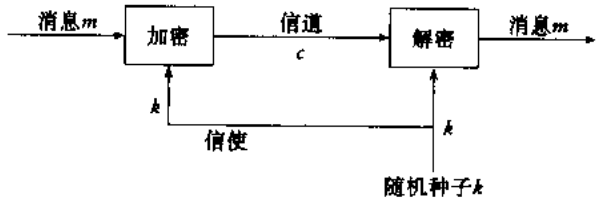
(2) 存储在密钥产生中心的特权信息的保密性；

(3) 在密钥产生中心给用户颁布 Smart 卡之前所完成的识别检测的严格性(要求用户提供的身份确实能唯一确定用户，而且用户不能否认)；

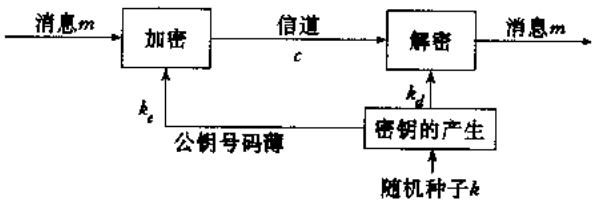
(4) 为了阻止用户的 Smart 卡的丢失、复制或未授权的使用，用户所采取的措施。

在基于身份的密码方案中，值得注意的是用户的秘密密钥必须由密钥产生中心来生成，不能由用户自己来生成，否则这个方案将是不安全的。

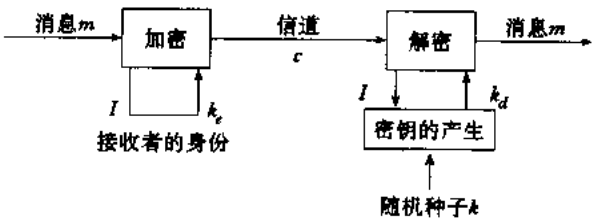
图 9.5.1 给出了私钥密码体制、公钥密码体制和基于身份的密码体制三者之间的差别。在这三种密码体制中，消息 m 用密钥 k_e 加密，密文 c 通过信道传送，并用密钥 k_d 解密，密钥的选择都基于真正的随机种子 k。在私钥密码体制中，$k_e=k_d=k$，密钥的保密和认证都需要使用单独的密钥信道(通常是一个信使(Courier))；在公钥密码体制中，加密密钥和解密密钥使用两个不同的函数来生成，即 $k_e=f_e(k)$，$k_d=f_d(k)$。在进行密钥认证时，需要



(a) 私钥密码体制



(b) 公钥密码体制

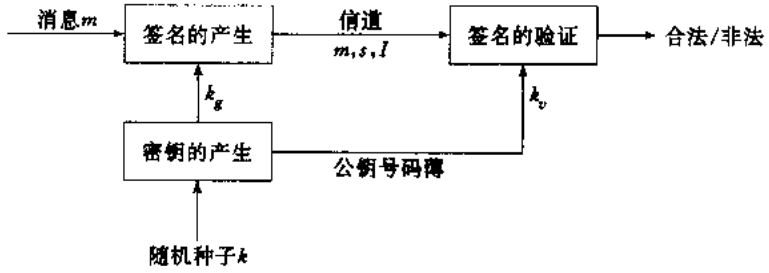


(c) 基于身份的密码体制

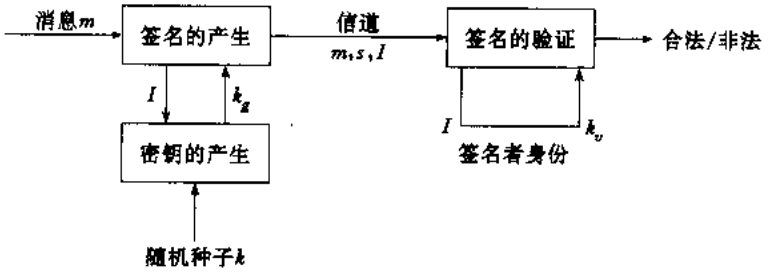
图9.5.1 三种密码体制之间的差别

有一个单独的信道(通常是一个薄子(Directory));在基于身份的密码体制中,加密密钥就是用户的身份,$k_e=I$,解密密钥由 I 和 k 导出,$k_d=f(I,k)$,这样,两个用户之间完全不需要独立的密钥信道。

图 9.5.2 给出了公钥签名方案和基于身份的签名方案两者之间的差别。用签名密钥 k_g 来对消息 m 签名,并将 m 及其签名 s 和签名者的身份 I 一起发送给验证者,验证者用验证密钥 k_v 来验证签名。



(a) 公钥签名方案



(b) 基于身份的签名方案

图9.5.2 公钥签名方案和基于身份的签名方案两者之间的差别

要实现 Shamir 的基于身份的密码方案的思想,我们需要一个具有下列两个附加条件的公钥密码体制:

(1)当知道种子 k 时,秘密密钥能从公钥中很容易地求出。

(2)从一对特定的公开密钥和秘密密钥求生成它们的种子 k 是困难的。

自从 Shamir 的基于身份的密码方案这一思想提出以来,人们围绕该话题进行了大量的研究,基于这一思想设计出的密码方案有很多。本小节我们仅介绍 Shamir 的基于身份的数字签名方案[20],下一小节将介绍 Guillou-Quisquater 的基于身份的识别协议和签名方案,对其它基于身份的密码方案感兴趣的读者请参阅文献[6,19,21,22,23,24]。

Shamir 的基于身份的数字签名方案[20]

该方案以下面的验证条件为基础:$s^e=it^{f(i,m)} \bmod n$,其中 m 为消息,s、t 是签名,i 是用户的身份,n 是两个大素数的乘积,e 是一个大素数,f 是一个单向函数。

参数 n、e 和单向函数 f 由密钥产生中心生成,所有用户都有同样的 n、e 和 f。这些值可以公开,但 n 的分解只有密钥产生中心知道。不同用户之间的唯一差别就是用户的身份 i 对应的秘密密钥 g,这里 $g^e \equiv i \pmod n$。g 可由密钥产生中心很容易的计算出来,但是如果 RSA 体制是安全的,那么没有别的人能求出 i 关于模 n 的 e 次根。

签名者 A 对消息 m 的签名过程为：

(1) 随机选择一个数 r，计算 $t=r^e \bmod n$ 和 $f(t,m)$；

(2) 计算 $s=gr^{f(t,m)} \bmod n$；

(3)A 对消息 m 的签名为(t,s)。

接收者 B 通过下式来验证(t,s)是否为 m 的签名：

$$s^e \equiv it^{f(t,m)} (\bmod n)。$$

9.5.2 Guillou-Quisquater 的基于身份的识别协议

Guillou-Quisquater 的基于身份的识别协议是由 Guillou-Quisquater 协议转化而来的。在这个方案中，TA 为用户 A 颁布一个数 u，该数是 A 的身份 ID 串的一个函数，而无需为用户 A 颁发一个证书。参数的选择与 Guillou-Quisquater 协议中的一样，这里 $ab \equiv 1 (\bmod \varphi(n))$。

TA 为 A 颁布一个值 u 的过程如下：

(1) TA 建立 A 的身份并颁布一个识别串 ID(A)；

(2) TA 计算 $u=(H(\mathrm{ID(A)})^{-1})^a \bmod n$ 并将 u 发送给 A。

其中 $H(.)$是一个公开的 Hash 函数。

Guillou-Quisquater 的基于身份的识别协议如下：

(1) A 随机选择一个数 k，$0 \leq k \leq n-1$，并计算 $\gamma=k^b \bmod n$；

(2) A 把 ID(A)和 γ 发送给 B；

(3) B 计算 $v=H(\mathrm{ID(A)})$；

(4) B 随机选择一个随机数 r，$0 \leq r \leq b-1$，并将 r 发送给 A；

(5) A 计算 $y=ku^r \bmod n$ 并将 y 发送给 B；

(6) B 验证 $\gamma=v^r y^b \bmod n$。

基于身份的 Guillou-Quisquater 签名方案[19]

签名者 A 选择两个大素数 p 和 q，形成 $n=pq$，并选择一个整数 e，使得 $\gcd(e,\varphi(n))=1$。选择一个整数 J_A，$1<J_A<n$，$\gcd(J_A,n)=1$，将 J_A 用作 A 的身份。J_A 的二元表示能被用来反映 A 的某些个人信息，诸如姓名、地址、驾驶执照号等等。A 按下列步骤确定一个整数 $a \in Z_n$，使得 $J_A a^e \equiv 1 (\bmod n)$，即计算 $a=(J_A^{-1})^{1/e} \bmod n$：

(1) 计算 $J_A^{-1} \bmod n$；

(2) 计算 $d_1=e^{-1} \bmod (p-1)$，$d_2=e^{-1} \bmod (q-1)$；

(3) 计算 $a_1=(J_A^{-1})^{d_1} \bmod p$，$a_2=(J_A^{-1})^{d_2} \bmod q$；

(4) 由中国剩余定理找到同余式方程组 $a \equiv a_1 (\bmod p)$，$a \equiv a_2 (\bmod q)$的一个解 a。A 公开 n、e、J_A，保密 a。另外 A 还选择一个单向 Hash 函数 H，并公开。签名者 A 对消息 m 的签名过程为：

(1) 随机选择一个整数 k，计算 $r=k^e \bmod n$；

(2) 计算 $l=H(m,r)$；

(3) 计算 $s=ka^l \bmod n$；

(4) A 对 m 的签名是对(l,s)。

接收者 B 验证签名的过程为：

(1) 获得 A 的公钥 n,e,J_A;

(2) 计算 $u=s^eJ_A^l \bmod n$ 和 $l'=H(m,u)$;

(3) 验证是否有 $l=l'$,如果 $l=l'$,则 B 接收 A 的签名,否则拒绝。

从识别协议构造数字签名的思想最早出现在文献[6]中,该文献也描述了一个基于身份的识别协议。

关于识别协议本章就介绍这么多。有关识别协议的综述性论文可参阅文献[25,26]。

9.6 注记和文献

在识别协议中,有两类协议是特别诱人的,一类是零知识识别协议,另一类是基于身份的识别协议。有关零知识识别协议的典型代表是 Feige-Fiat-Shamir 识别协议,参见文献[7]。有关基于身份的识别协议是 Shamir 首次提出的一种观点,参见文献[20]。

参 考 文 献

[1] Evans, A. Kantrowitz, w. and Weiss, E., A User Identification Scheme Not Requiring Secrecy in the Computer, Communications of the ACM, Vol. 17, No. 8, Aug. 1974, pp. 437—472.

[2] Morris, R. and Thompson, K., Password Security: A Case History, Communications of the ACM, Vol. 22, No. 11, Nov. 1979, pp. 594—597.

[3] Purdy, G. P., A High-security log-in Procedure, Communications of the ACM, Vol. 17, No 8, Aug, 1976, pp. 442—445.

[4] Feldmeier, D. C. and Karn, P. R., Unix Password Security —— Ten Years Later, Advances in Cryptology Crypto'89, Springer-Verlag, 1990, pp. 44—63.

[5] Klein, D. V., Foiling the Cracker: A Survey of, and Implications to, Password Security, Proceedings of the USENIX UNIX Security Workshop, Aug. 1990., pp. 5—14.

[6] Fint, A. and, Shamir, A., How to Prove Yourself: Practical Solutions to Identification and Signature Problems, Advances in Cryptology-Crypto'86, Springer-Verlag, 1987, pp. 186—194.

[7] Feige, U., Fiat, A. and Shamir, A., Zero Knowledge Proofs of Identity, proceedings of STOC, 1987, pp. 210—219

[8] Shamir, A. and, Fiat, A., Apparatus and Article for Identification and Signature, U. S. Patent 4, 748, 668, 31 May 1988.

[9] Laudau, S, Zero-knowledge and the Department of Defense, Notices of the American Mathematical Society, Vol. 35, No. 1, Jan. 1988, pp. 5—12.

[10] Micali, S. and Shamir, A, An Improvement on the Fiat-Shamir Identification and Signature scheme, Advances in Cryptology-Crypto'88, Springer-Verlag, 1990, pp. 244—247.

[11] Brickell, E. F., Lee, P. J. and Yacobi, Y., Secure Audio Teleconference, Advances in Cryptology-Crypto'87, Springer-Verlag, 1988, pp. 418—426.

[12] Ong, H. and Schnorr, C. P., Fast Signature Generation with a Fiat Shamir-like Scheme, Advances in Cryptology-Eurocrypt'90, Springer-Verlag, 1991, pp. 432—440.

[13] Schnorr, C. P., Efficient Signature Generation for Smart Cards, Journal of Cryptology, Vol. 4, No3, 1991, pp. 161—174.

[14] Chaum, D., Evertse J. H. and, Van de Graff, J., An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some Generalizations, Advances in Cryptology-Eurocrypt'87, Springer-Verlag, 1988, pp. 127—141.

[15] de Rooij, P., On the Security of the Schnorr Scheme Using Preprocessing, Advances in Cryptology-Eurocrypt'91, Springer-Verlag, 1991, pp. 71—80.

[16] Brickell, E. F. and McCurley, K. S., An Interactive Identification Scheme Based on Discrete Logarithms and Factoring, Advances in Cryptology-Eurocrypt'90, Springer-Verlog, 1991, pp. 63—71.

[17] Okamoto, T., Provably Secure and Practical Identification Schemes and Corresponding Signature Scheme, Advances in Cryptology-Crypto'92, Springer-Verlog, 1993, pp. 31—53.

[18] Guillou, L. C. and Quisquater, J. J., A Practical Zero-knownledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1989, pp. 123—128.

[19] Guillou, L. C. and Quisquater, J. J., A Paradoxical Identity-based Signature Scheme Resulting from Zero-knownledge, Advances in Cryptology-Crypto'88, Springer-Verlag, 1990, pp. 216—231.

[20] Shamir, A., Identity-based Cryptosystems and Signature Schemes, Advances in Cryptology-Crypto'84, Springer-Verlag, 1985, pp. 47—53.

[21] Harn, L. and Yang, S. B., ID-based Cryptographic Schemes for User Identification, Digital Signature, and Key Distribution, IEEE J. Select, Areas Commun, Vol. 71, pp. 757—760.

[22] Tanaka, H., A Realization Scheme for the Identity-based Cryptosystem, Advances in Crytology-Crypto'87, Springer-Verlag, 1988, pp. 340—349.

[23] 陶仁骐，陈世华，基于身份的密码体制和数字签名的有限自动机公开钥密码实现，密码学进展-ChinaCrypt'92，科学出版社，北京，87—104 页.

[24] Tsujii, S. and Chao, J., A New ID-based Key Sharing System, Advances in Cryptology-Crypto'91, Springer-Verlag, 1992, pp. 288—299.

[25] Burmester, M., Desmedt, Y. and Beth, T., Efficient Zero-knowledge Identification Schemes for Smart Cards, The Computer Journal, 35(1992), pp. 21—29.

[26] De Waleffe, D. and Quisquater, J. J., Better Login Protocols for Computer Networks, Computer Security and Industrial Cryptography (State of the Art and Evolution, ESAT Course, May. 1991), Springer-Verlag, 1993, pp. 50—70.

[27] Brickell, E. F. and Mccurley, K. S., An Interactive Identification Scheme Based on Discrete Logarithms and Factoring, Journal of Cryptology, 5(1992), pp. 29—39.

[28] Guillou, L. C., Ugon, M. and Quisquater, J. J., The Smart Card: A Standardized Security Device Dedicated to Public Cryptology, Contemporary Cryptology. The Science of Information Integrity, Piscatoway, N. J., IEEE press, 1992, pp. 561—613.

[29] 冯登国、裴定一，一个新型认证方案的设计与分析，计算机工程与应用，Vol. 33, No. 2, 1997, 50—51 页.

[30] Stinson, D. R., Cryptography-Theory and Practice, CRC Press, 1995.

[31] Menezes, A. J., Van Oorschot, P. C. and Vanstone, S. A., Handbook of Applied Cryptography, IEEE Press, 1996.

第 10 章　密钥管理技术

在 Kerckhoff 假设下，一个密码系统的安全性取决于对密钥的保护，而不是对系统或硬件本身的保护。密码体制可以公开，密码设备可能丢失，同一型号的密码机仍可继续使用。然而一旦密钥丢失或出错，不但合法用户不能提取信息，而且可能会使非法用户窃取信息。可见，密钥的保密和安全管理在数据系统安全中是极为重要的。

密钥管理包括密钥的产生、存贮、装入、分配、保护、丢失、销毁以及保密等内容。其中分配和存贮可能是最棘手的问题。密钥管理不仅影响系统的安全性，而且涉及到系统的可靠性、有效性和经济性。当然，密钥管理过程中也不可能避免物理上、人事上、规程上等一些问题。但本章将主要从理论和技术上讨论密钥管理的有关问题。

10.1　密钥的种类和密钥的生成、装入

10.1.1　密钥的种类

密钥的种类很多，但主要有下述几种：

(1) 基本密钥(base key)，又称初始密钥(primary key)，以 k_p 表示，是由用户选定或由系统分配给用户的可在较长时间(相对于会话密钥)内由一对用户所专用的秘密密钥，故又称用户密钥(user key)。要求它既要安全，又要便于更换，和会话密钥一起去启动和控制某种算法构造的密钥生成器，来产生用于加密数据的密钥流，参见图 10.1.1。

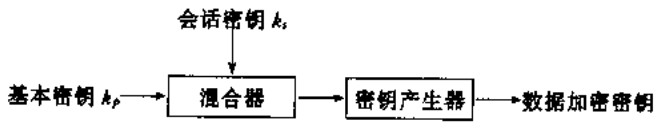


图 10.1.1　几种密钥之间的关系

(2) 会话密钥(session key)。两个通信终端用户在一次通话或交换数据时所用的密钥，以 k_s 表示。当用其对传输的数据进行保护时称为数据加密密钥(data encrypting key)，当用它保护文件时称为文件密钥(file key)。会话密钥的作用是使我们可以不必太频繁地更换基本密钥，有利于密钥的安全和管理方便。这类密钥可由用户双方预先约定，也可由系统动态地产生并赋予通信双方，它为通信双方专用，故又称专用密钥(private key)。

(3) 密钥加密密钥(key encrypting key)。用于对传送的会话或文件密钥进行加密时采用的密钥，也称次主密钥(submaster key)或辅助(二级)密钥(secondary key)，以 k_e 表示。通信网中每个节点都分配有一个这类密钥。为了安全，各节点的密钥加密密钥应互不相同。在主机和主机之间以及主机和各终端之间传送会话密钥时都需要有相应的密钥加密密钥。每台主机都须存贮有关至其它各主机和本主机范围内各终端所用的密钥加密密钥，而各终端只需要一个与其主机交换会话密钥时所需的密钥加密密钥，称之为终端主密钥(terminal master key)。在主机和一些密码设备中，存贮各种密钥的装置应有断电保护

和防窜扰、防欺诈等控制功能。

(4) 主机主密钥(host master key)。它是对密钥加密密钥进行加密的密钥，存于主机处理器中，以 k_m 表示。

除了上述几种密钥之外，还有：用户选择密钥(custom option key)，用来保证同一类密码机的不同用户可使用的不同的密钥；族密钥(family key)及算法更换密钥(algorithm changing key)等。这些密钥的主要作用是在不增大更换密钥工作量的条件下，扩大可使用的密钥量。基本密钥一般通过面板开关或键盘选定，而用户选择密钥常要通过更改设备内部连线，改变插接组件或开关位置实现，这种改变常起到更改密钥产生算法的作用。例如，在非线性移存器密钥流产生器中，基本密钥和会话密钥用于确定寄存器的初态，而用户选择密钥可决定寄存器反馈线抽头的连接。

10.1.2 密钥的生成

现代通信网需要产生大量的密钥分配给各主机、节点和用户。过去靠人工方法产生和管理密钥的方式已被时代所淘汰。密钥产生和管理的自动化，不仅可以减轻人们的工作负担，而且可以消除人为差错引起的泄密。目前已有各种各样的密钥产生器可为大型系统提供所需的各类密钥。生成密钥的算法要满足一定的要求才能启用。

(1) 主机主密钥的产生。这类密钥通常要用诸如掷硬币、骰子，从随机数表中选数等随机方式产生，以保证密钥的随机性，避免可预测性。而任何机器和算法所产生的密钥都有被预测的危险。主机主密钥是控制产生其它加密密钥的密钥，而且长时间保持不变，因此它的安全性是至关重要的。

(2) 密钥加密密钥的产生，可以由机器自动产生，也可以由密钥操作员选定。密钥加密密钥构成的密钥表存贮在主机中的辅助存贮器中，只有密钥产生器才能对此表进行增加、修改、删除和更换密钥，其副本则以秘密方式送给相应的终端或主机。一个有 N 个终端用户的通信网，若要求任一对用户之间彼此能进行保密通信，则需要有 C_N^2 个密钥加密密钥。当 N 较大时，难免有一个或数个被敌手掌握。因此，密钥产生算法应当能保证其它用户的密钥加密密钥仍有足够的安全性。可用随机比特产生器(如噪声二极管振荡器等)或伪随机数产生器生成这类密钥，也可用主密钥控制下的某种算法来产生[1]。

(3) 会话密钥的产生。会话密钥可在密钥加密密钥作用下通过某种加密算法动态地产生，如用初始密钥控制一非线性移存器或用密钥加密密钥控制 DES 算法产生。

初始密钥可用产生密钥加密密钥或主机主密钥的方法生成。

10.1.3 密钥的装入

密钥的装入是指将密钥装入密码装置内的过程。

(1)主机主密钥的装入。为了安全，主机主密钥一旦装入后就不再读取了。但密码管理人员可在非常安全的条件下证实密码机中的主密钥就是装入的那个主密钥。例如，选一个随机数 RN，并以主密钥 k_m 对 RN 加密得 $E_{k_m}(\mathrm{RN})$。寻求一个函数 ϕ，使得 $E_{k_m}(\mathrm{RN})=\phi(k_m)$，记住 $\phi(k_m)$ 及 RN。当主密钥装入后，通过用它对随机数 RN 加密并与 $\phi(k_m)$ 比较就可验证装入的主密钥是否正确无误。可用乒乓开关、拨号盘、键盘、穿孔卡、磁带机等硬件输入主密钥，这些硬件必须有良好的电磁屏蔽。为了防止人为和机械错误可采用奇偶校验等

措施。主密钥也可用间接法装入，先将其读入主处理器的主存器中，而后通过置主密钥操作将其写入密码器中的非易失存贮器中。

(2) 终端主密钥的装入。可用面板开关、拨号盘、袖珍密钥注入器、键盘等装入。装入后也不能再读取，需要通过检验来证实。可用联机法检验，即终端和主机进行一次会话，看是否能正常通信。也可用脱机检验，用主密钥加密、数据加密和数据脱密等操作，实现数据加密和脱密全过程，看是否能正常工作。

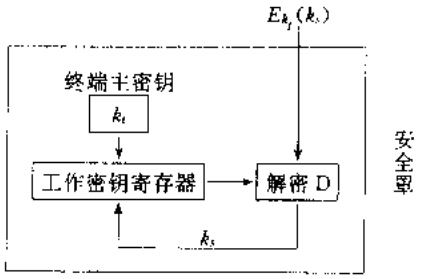


图 10.1.2 终端会话密钥的装入

(3) 终端会话密钥的装入，可通过图 10.1.2 的解密过程实现。主机将加密的会话密钥 $E_{k_t}(k_s)$ 通过线路传至终端，终端用终端主密钥 k_t 解密得到会话密钥 k_s，将其送至工作(正作用的)密钥寄存器中以备对加密的数据进行解密。

10.2 密钥分配协议

密钥分配(key distribution)是这样的一种机制：系统中的一个成员先选择一个秘密密钥，然后将它传送给另一个成员或别的成员。传统的方法是通过邮递或信使护送密钥。密钥可用打印、穿孔纸带或电子形式记录。这种方法的安全性完全取决于信使的忠诚和素质，需要精心挑选信使，但很难完全消除信使被收买的可能性。这种方法成本很高，薪金不能低，否则，会危及安全性，有人估计此项支出可达整个密码设备费用的三分之一。这种方法一般可保证及时性和安全性，偶尔会出现丢失、泄密等。为了减少费用可采用分层方式，信使只传送密钥加密密钥，而不去传送大量的数据加密密钥，既减少了信使的工作量(因而大大降低了费用)又克服了用一个密钥加密过多数据的问题。当然，这不能完全克服信使传送密钥的弱点。另外，如果网络中有 n 个用户，则这种方法需要 C_n^2 个安全信道。但借助一个可信中心可将安全信道从 C_n^2 降低到 n 个。可信中心的作用是验证用户的身份，给用户选择和传输密钥等。对每一对用户 U 和 V，可信中心选择一个随机密钥 $K_{U,V}=K_{V,U}$，将它在一个安全的信道上传送给 U 和 V。因为可信中心和网络中的每个用户都必须有一个安全信道，所以在有 n 个用户的网络中共需 n 个安全信道。实际上，这样做也不切实际，这是因为每个用户必须存贮 $n-1$ 个密钥，TA 需要安全地传送 C_n^2 个密钥。当 n 稍大时，传输量和存贮量都很大。可见，我们设计密钥分配协议时必须考虑下列两个因素：

(1) 传输量和存贮量都尽可能地小；

(2) 每一对用户 U 和 V 都能独立地计算一个秘密密钥 $K_{U,V}$。

下面我们来介绍三个密钥分配协议。

10.2.1 Blom 方案

假定我们有一个有 n 个用户的网络。为方便起见，我们假定密钥从 Z_p 中选择，$p\geqslant n$ 是一个素数。设 k 是一个整数，$1\leqslant k\leqslant n-2$。$k$ 是使得网络中的任何 k 个用户合伙方案仍是安全的最大值。在 Blom 方案中[127](也称规模为 k 的 Blom 方案)，可信中心给每个用户在

一个安全的信道上将发送 Z_p 中的 $k+1$ 个元素。每一对用户 U 和 V 将能计算一个密钥 $K_{U,V}=K_{V,U}$。安全性条件为：至多 k 个不同用户的任何用户集必须不能确定关于 $K_{U,V}$ 的任何信息(注意这里所说的安全性是指无条件安全性)。我们首先提出 Blom 方案的一个特殊情况，即 $k=1$(规模为 1 的 Blom 方案)。此时，可信中心给每个用户在一个安全的信道上将发送 Z_p 中的两个元素。任何单个用户 $W(\neq U,V)$不能确定关于 $K_{U,V}$ 的任何信息。下面我们来描述 Blom 密钥分配方案($k=1$)。

Blom 密钥分配方案

(1) 公开一个素数 p，每个用户 U 公开一个元素 $r_U\in Z_p$，这些元素 r_U 必须互不相同。

(2) 可信中心选择三个随机元素 $a,b,c\in Z_p$(未必不同)，并且形成多项式：

$$f(x,y)=(a+b(x+y)+cxy) \bmod p$$

(3) 对每一个用户 U，可信中心计算多项式：$g_U(x)=f(x,r_U) \bmod p$ 并将 $g_U(x)$ 在一个安全的信道上传送给 U。注意 $g_U(x)$ 是 x 的一个线性函数，所以它可以写为：$g_U(x)=a_U+b_U x$，这里 $a_U=(a+br_U) \bmod p$，$b_U=(b+cr_U) \bmod p$。

(4) 如果 U 和 V 想通信，那么他们使用共同密钥

$$K_{U,V}=K_{V,U}=f(r_U,r_V)=(a+b((r_U+r_V)+cr_Ur_V) \bmod p$$

这里 U 计算 $K_{U,V}=f(r_U,r_V)=g_U(r_V)$，$V$ 计算 $K_{V,U}=f(r_V,r_U)=g_V(r_U)$。

我们现在证明，没有一个用户能确定关于另两个用户的密钥的任何信息。

定理 10.2.1 $k=1$ 时的 Blom 方案对任何单个用户是无条件安全的。

证明：让我们假定用户 W 想试图计算密钥：

$$K_{U,V}=(a+b(r_U+r_V)+cr_Ur_V) mod\ p$$

值 r_U 和 r_V 是公开的，但值 a,b 和 c 是未知的。因为可信中心将 $g_W(x)$ 通过一个安全的信道发送给了 W，所以，W 知道值 $a_W=(a+br_W) \bmod p$ 和 $b_W=(b+cr_W) \bmod p$。

我们下面来说明 W 知道的信息和密钥 $K_{U,V}$ 的任何可能的取值 $l\in Z_p$ 一样。因此，W 不能排除 $K_{U,V}$ 的任何值。考虑下列的矩阵方程：

$$\begin{bmatrix}1 & r_U+r_V & r_Ur_V\\ 1 & r_W & 0\\ 0 & 1 & r_W\end{bmatrix}\begin{bmatrix}a\\ b\\ c\end{bmatrix}=\begin{bmatrix}l\\ a_W\\ b_W\end{bmatrix}$$

第一个方程来自假定 $K_{U,V}=l$，第二个和第三个方程来自 W 从 $g_W(x)$ 知道的关于 a,b 和 c 的信息。系数矩阵的行列式为

$$r_W^2+r_Ur_V-(r_U+r_V)r_W=(r_W-r_U)(r_W-r_V)$$

这里的所有运算都是 Z_p 上的运算。因为 $r_W\neq r_U,r_V$，所以，上述矩阵方程关于 a,b,c 有唯一的一个解。换句话说，$K_{U,V}$ 的任何可能的值 l 和 W 知道的信息一样。

两个用户，比方说 W 和 X 合伙便能确定任何密钥 $K_{U,V}$，这里 $\{W,X\}\cap\{U,V\}=\varnothing$。$W$ 和 X 知道

$$\begin{cases}a_W=a+br_W\\ b_W=b+cr_W\\ a_X=a+br_X\\ b_X=b+cr_X\end{cases}$$

这样，有四个方程式三个未知量，用户就可容易地计算出 a，b 和 c 的值。一旦用户知道 a，b 和 c 的值，他们就能形成多项式 $f(x,y)$并能计算他们所希望的任何密钥。

规模为 1 的 Blom 方案可直接推广为规模为 $k(1\leqslant k\leqslant n-2)$的 Blom 方案。唯一需要改变的就是第(2)步。可信中心将使用一个如下形式的多项式：

$$\sum_{i=0}^{k}\sum_{j=0}^{k}a_{ij}x^iy^j \bmod p$$

这里，$a_{ij}\in Z_p(0\leqslant i\leqslant k,0\leqslant j\leqslant k)$，并且对所有的 $i,j,a_{ij}=a_{ji}$。协议的其余步骤不需要改变。

另外，对一般化的 Blom 方案感兴趣的读者请参阅文献[3,4]。

10.2.2 Diffie-Hellman 密钥预分配方案

Diffie-Hellman 密钥预分配方案[5]是下一节将要介绍的 Diffie-Hellman 密钥交换协议[6,7]的一种修改。该方案在假定与离散对数问题相关的一个问题是难处理的情况下是计算上安全的。

我们仅描述在 Z_p 上的一个方案，p 是一个素数。不过，该方案可在计算离散对数问题是难处理的任何有限群上实现。假定 α 是 Z_p 的一个本原元，网络中的任何用户都知道 p 和 α 的值。

在这个方案中，我们用 ID(U)表示网络中用户 U 的某些识别信息，诸如姓名、E-mail 地址、电话号码或别的有关信息。每个用户 U 有一个秘密指数 $a_U(0\leqslant a_U\leqslant p-2)$和一个相应的公钥 $b_U=\alpha^{a_U}\bmod p$。

可信中心有一个签名方案，该签名方案的公开验证算法记为 Ver_{TA}，秘密签名算法记为 Sig_{TA}。我们暗含着在签名消息之前，先将消息用一个公开的 Hash 函数杂凑。但为了叙述简单起见，我们在这里略去这一步。

当一个用户 U 入网时，可信中心需给他颁发一个证书(certificate)。用户 U 的证书为：$C(U)=(\mathrm{ID}(U),b_U,\mathrm{Sig}_{TA}(\mathrm{ID}(U),b_U))$。可信中心无需知道 a_U 的值。证书可存贮在一个公开的数据库中，也可由用户自己存贮。可信中心对证书的签名允许网络中的任何人能验证它所包含的信息。U 和 V 计算共同密钥 $k_{U,V}=\alpha^{a_Ua_V}\bmod p$ 的协议即 Diffie-Hellman 密钥预分配协议如下：

(1)公开一个素数 p 和一个本原元 $\alpha\in Z_p^*$。

(2)V 使用他自己的秘密值 a_V 及从 U 的证书中获得的公开值 b_U，计算 $k_{U,V}=\alpha^{a_Ua_V}\bmod p=b_U^{a_V}\bmod p$。

(3)U 使用他自己的秘密值 a_U 及从 V 的证书中获得的公开值 b_V，计算 $k_{U,V}=\alpha^{a_Ua_V}\bmod p=b_V^{a_U}\bmod p$。

现在我们来考虑一下 Diffie-Hellman 密钥预分配方案的安全性。可信中心对用户的证书的签名有效地阻止了敌手 W 改变别人的证书的任何信息，从而阻止了 W 的主动攻击。因此，我们关心的是 W 的被动攻击。这样，与此相关问题就是一个用户 $W(\neq U,V)$能否计算 $k_{U,V}$？换句话说，给定 $\alpha^{a_U}\bmod p$ 和 $\alpha^{a_V}\bmod p$，但不知道 a_U 和 a_V，计算 $\alpha^{a_Ua_V}\bmod p$ 是否可行？这个问题称为 Diffie-Hellman 问题。Diffie-Hellman 问题可等价地陈述为：给定 p,α,β 和 γ，p 为一个素数，$\alpha\in Z_p^*$ 是一个本原元，$\beta,\gamma\in Z_p^*$，计算 $\beta^{\log_\alpha\gamma}\bmod p\ (=\gamma^{i\log_\alpha\beta}\bmod p)$。显然，Diffie-Hellman 密钥预分配方案对一个被动的敌手是安全的，当且仅当 Diffie-Hellman

问题是难处理的。

如果 W 能从 b_U 确定 a_U，或从 b_V 确定 a_V，那么他一定能像 U 或 V 一样计算出 $k_{U,V}$。但我们假定 Z_p 上的离散对数问题是难处理的，所以对 Diffie-Hellman 密钥预分配方案的这种类型的攻击是计算上不可行的。人们猜测解 Diffie-Hellman 问题的任何算法也可用来解离散对数问题。但至今这个猜测还未得到证明。正像讨论 RSA 体制的安全性一样，人们猜测但未被证明，破译 RSA 体制多项式等价于因子分解。

虽然我们不能精确地说 Diffie-Hellman 问题有多难，但我们可以将这个问题和我们已讨论过的 ElGamal 体制的安全性联系起来。

定理 10.2.2 破译 ElGamal 体制等价于解 Diffie-Hellman 问题。

证明：ElGamal 体制的秘密密钥为 a，公钥 $\beta=\alpha^a \bmod p$，p 和 α 是公开知道的，p 是一个素数，α 是 Z_p^* 的一个本原元。

对消息 $x\in Z_p$ 的加密过程为：随机选择一个数 $k\in Z_{p-1}$，$E_K(x,k)=(y_1,y_2)$，这里 $y_1=\alpha^k \bmod p$，$y_2=x\beta^k \bmod p$。

解密过程为

$$D_K(y_1,y_2)=y_2(y_1^a)^{-1} \bmod p \qquad y_1,y_2\in Z_p^*。$$

假定我们有一个算法 A，A 能解 Diffie-Hellman 问题，并给定一个用 ElGamal 体制加密的密文 (y_1,y_2)。我们将 p,α,y_1 和 β 作为算法 A 的输入，则我们获得 $A(p,\alpha,y_1,\beta)=A(p,\alpha,\alpha^k,\alpha^a)=\alpha^{ka} \bmod p=\beta^k \bmod p$。此时密文 (y_1,y_2) 对应的明文可由下式容易求出：$x=y_2(\beta^k)^{-1} \bmod p$

反之，假定我们有一个算法 B，B 能完成 ElGamal 体制的解密过程。也就是说，如果将 p,α,β,y_1 和 y_2 作为 B 的输入，那么我们能求得 $x=y_2(y_1^{\log_\alpha^\beta})^{-1} \bmod p$。现在给定 p,α,β 和 γ，p 为一个素数，$\alpha\in Z_p^*$ 是一个本原元，$\beta,\gamma\in Z_p^*$，则我们可利用下列办法计算出 $\gamma^{\log_\alpha^\beta}$：$B(p,\alpha,\beta,\gamma,1)^{-1}=(1(\gamma^{\log_\alpha^\beta})^{-1})^{-1} \bmod p=\gamma^{\log_\alpha^\beta} \bmod p$。

10.2.3 Kerboros 协议

在前面讨论的密钥预分配方法中，每对用户能计算一个固定的密钥。如果同一个密钥使用的周期长了以后，那么就可能存在伪造的危险。因此，人们更喜欢使用在线方法分配密钥，也就是一对用户想通信时每次产生一个新的会话密钥。

如果使用在线密钥分配，那么网络中的每个用户和可信中心共享一个密钥而无需再存贮别的密钥。会话密钥将通过请求可信中心来传送。确保密钥新鲜(key freshness)是可信中心的职责。

Kerboros 协议是一个基于私钥密码体制的流行的密钥服务系统。在这个系统中，每个用户 U 和可信中心共享一个秘密的 DES 密钥。在 Kerboros 协议(版本 V)的最新版本中，传送的所有的消息都通过 CBC 模式(这种模式参见 5.4 节)进行加密。

象上一小节一样，我们用 ID(U)表示用户 U 的某些识别信息。使用 Kerboros 协议传输一个会话密钥的过程可描述如下：

(1)用户 U 为了和用户 V 通信，他向可信中心要一个会话密钥；

(2)可信中心随机选择一个会话密钥 K、一个时戳 T 和一个生存期 L；

(3)可信中心计算 $m_1=E_{K_U}(K,\mathrm{ID}(V),T,L)$ 和 $m_2=E_{K_V}(K,\mathrm{ID}(U),T,L)$，并将 m_1 和 m_2 发送给 U；

(4)用户 U 首先解密 m_1 获得 $K,\mathrm{ID}(V),T$ 和 L。然后计算 $m_3=E_K(\mathrm{ID}(U),T)$ 并将 m_3 和可信中心发送来的 m_2 一起发送给 V；

(5)用户 V 首先解密 m_2 获得 $K,\mathrm{ID}(U),T$ 和 L。然后使用 K 解密 m_3 获得 T 和 $\mathrm{ID}(U)$ 并检测 T 的两个值和 $\mathrm{ID}(U)$ 的两个值是否一样。如果是一样的，那么 V 计算 $m_4=E_K(T+1)$，并将 m_4 发送给 U；

(6)U 使用 K 解密 m_4 获得 $T+1$ 并验证解密结果是 $T+1$。

传输在协议中的信息可图示为

$$\text{可信中心}\xrightarrow[E_{K_V}(K,\mathrm{ID}(U),T,L)]{E_{K_U}(K,\mathrm{ID}(V),T,L)}\text{用户 }U\xrightarrow[E_K(T+1)]{E_K(\mathrm{ID}(U),T)\ E_{K_V}(K,\mathrm{ID}(U),T,L)}\text{用户 }V$$

现在我们对协议中的各步骤作一些解释。在第(2)步，可信中心产生 K,T 和 L。在第(3)步，可信中心使用它与用户 V 共享的密钥 K_U 加密 K,T,L 和 $\mathrm{ID}(V)$ 形成 m_1。使用它与用户 V 共享的密钥 K_V 加密 K,T,L 和 $\mathrm{ID}(U)$ 形成 m_2。将这些加密消息发送给用户 U。U 能使用他的密钥解密 m_1，获得 K,T,L 和 $\mathrm{ID}(V)$。他将验证目前的时间是在区间 $[T,T+L]$ 内。他通过验证解密获得的 $\mathrm{ID}(V)$ 检测可信中心颁布给他的密钥 K 是他和用户 V 的会话密钥。用户 U 将 m_2 转发给 V，并且 U 将使用新的会话密钥 K 加密 T 和 $\mathrm{ID}(U)$，将加密结果 m_3 发送给 V。当用户 V 从 U 那里收到 m_2 和 m_3 时，他解密 m_2 获得 T,K,L 和 $\mathrm{ID}(U)$，然后使用 K 解密 m_3 获得 T 和 $\mathrm{ID}(U)$，并验证两个 T 值和两个 $\mathrm{ID}(U)$ 值一样。这可使 V 相信加密在 m_2 中的会话密钥和用于加密 $T,\mathrm{ID}(U)$ 的密钥是一样的。V 使用 K 加密 $T+1$，并将所得的结果 m_4 送回 U。当 U 收到 m_4 时，他使用 K 解密并验证解密结果是 $T+1$。这可使 U 相信会话密钥 K 已经被成功地传输给了 V，因为产生消息 m_4 时需要 K。消息 m_1 和 m_2 用来提供会话密钥 K 在传输过程中的秘密性。m_3 和 m_4 用来提供密钥确证性(key confirmation)，也就是能使 U 和 V 相互相信他们拥有同样的会话密钥 K。在大多数密钥分配方案中，都可实现密钥确证性这一特性。一般地，可类似于 Kerboros 协议中的做法来实现这一特性，即通过使用新的会话密钥 K 加密已知的量。在 Kerboros 协议中，U 使用 K 加密 $\mathrm{ID}(U)$ 和 T。V 使用 K 加密 $T+1$。

时戳 T 和生存期 L 的目的是阻止一个主动的敌手存贮旧消息并在后来重发。这种攻击称为重放攻击(replay attack)。这种方法之所以奏效是因为一旦密钥过期，它不能再被作为合法的密钥接收。

Kerboros 协议的缺点之一是网络中的所有用户都得同步时钟，因为目前的时间被用来确定是否一个给定的会话密钥 K 是合法的。在实际中，提供完全的同步是很困难的，所以允许有一定量的时差。

有关 Kerboros 协议的一些最新发展，感兴趣的读者请参阅文献[8]。

10.3 密钥协定

密钥协定(key agreement)是一个协议，它通过两个或多个成员在一个公开的信道上通信联合地建立一个秘密密钥。在一个密钥协定方案中，密钥的值是由两个成员提供的输入的一个函数。第一个密钥协定协议就是众所周知的 Diffie-Hellman 密钥交换协议。我们假定 p 是一个素数，α 是 Z_p 的一个本原元，p 和 α 是公开知道的。Diffie-Hellman 密钥交换协议与上节描述的 Diffie-Hellman 密钥预分配协议很相似。二者的差别在于：Diffie-Hellman 密钥交换协议中的用户 U 和 V 的指数 a_U 和 a_V 不断地在更新而不是固定的。Diffie-Hellman 密钥交换协议[6,7]可描述为：

(1) U 随机地选择 a_U，$0 \leqslant a_U \leqslant p-2$；

(2) U 计算 $\alpha^{a_U} \bmod p$ 并发送给 V；

(3) V 随机地选择 a_V，$0 \leqslant a_V \leqslant p-2$；

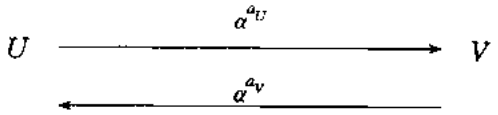
(4) V 计算 $\alpha^{a_V} \bmod p$ 并发送给 U；

(5) U 计算 $K=(\alpha^{a_V})^{a_U} \bmod p$，$V$ 计算 $K=(\alpha^{a_U})^{a_V} \bmod p$。

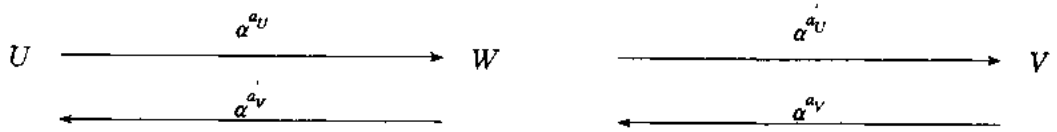
下面我们再介绍三个密钥协定协议。第一个是 Diffie-Hellman 密钥交换协议的一种变形，另两个是 MTI 方案和 Girault 方案。

10.3.1 端-端协议

Diffie-Hellman 密钥交换的基本模式为



不幸的是，该协议易受一个主动攻击者进行中间入侵(intruder-in-middle)攻击。一个主动攻击者 W 进行中间入侵攻击的基本模式为



在协议末，U 实际上和 W 建立了秘密密钥 $\alpha^{a_U a_V'}$，V 和 W 建立了秘密密钥 $\alpha^{a_V a_U'}$。当 U 加密一个消息发送给 V 时，W 能解密它而 V 不能。类似地，当 V 加密一个消息发送给 U 时，W 能解密它而 U 不能。

显然，为了克服中间入侵攻击，必须确保用户 U 和 V 正在交换消息而不是和 W 交换消息。在交换密钥之前，U 和 V 可以通过完成一个协议来建立相互的身份，例如可通过使用第 9 章介绍的识别协议，但这仍不能抵抗中间入侵攻击。这是因为 W 可以等到 U 和 V 相互证明身份之后进行中间入侵攻击。因此，在密钥建立的同时，密钥协定协议应能自己认证参加者的身份。这种协议称为认证密钥协定(authenticated key agreement)。我们将描述一个认证密钥协定协议，该协议称为端-端协议(station-to-station protocol)，它是 Diffie-Hellman 密钥交换协议的一个修改。

假定 p 是一个素数，α 是 Z_p 的一个本原元，p 和 α 是公开知道的。每个用户 U 有一个

签名方案，将其签名算法记为 Sig_U，验证算法记为 Ver_U。可信中心也有一个签名方案，它的公开验证算法为 Ver_{TA}。每个用户有一个证书 $C(U)=(\mathrm{ID}(U),\mathrm{Ver}_U,\mathrm{Sig}_{TA}(\mathrm{ID}(U),\mathrm{Ver}_U))$，这里 $\mathrm{ID}(U)$ 是 U 的识别信息。端-端协议的工作过程如下(这里是一个简化了的端-端协议)：

(1) U 随机选择一个数 a_U，$0\leqslant a_U\leqslant p-2$；

(2) U 计算 $\alpha^{a_U}\bmod p$ 并发送给 V；

(3) V 随机地选择一个数 a_V，$0\leqslant a_V\leqslant p-2$；

(4) V 先计算 $\alpha^{a_V}\bmod p$，然后计算 $K=(\alpha^{a_U})^{a_V}\bmod p$ 和 $y_V=\mathrm{Sig}_V(\alpha^{a_U},\alpha^{a_V})$；

(5) V 将 $(C(V),\alpha^{a_V},y_V)$ 发送给 U；

(6) U 计算 $K=(\alpha^{a_V})^{a_U}\bmod p$，他使用 Ver_V 验证 y_V，使用 Ver_{TA} 验证 $C(V)$；

(7) U 计算 $y_U=\mathrm{Sig}_U(\alpha^{a_U},\alpha^{a_V})$ 并将 $(C(U),y_U)$ 发送给 V；

(8) V 使用 Ver_U 验证 y_U，使用 Ver_{TA} 验证 $C(U)$。

在简化的 STS 协议中，交换的信息(不包括证书)可图示为

$$U \xrightarrow{\alpha^{a_U}} \xleftarrow[\mathrm{Sig}_U(\alpha^{a_U},\alpha^{a_V})]{\alpha^{a_V},\mathrm{Sig}_V(\alpha^{a_U},\alpha^{a_V})} V \xrightarrow{\quad}$$

下面我们看一看该协议如何来抵抗中间入侵攻击。当 W 截取 α^{a_U} 后用 α^{a_U} 来代替 α^{a_U} 时，W 从 V 那里收到 α^{a_V}，$\mathrm{Sig}_V(\alpha^{a_U},\alpha^{a_V})$。他希望用 α^{a_V} 来代替 α^{a_V}。然而，这个意味着他必须也能用 $\mathrm{Sig}_V(\alpha^{a_U},\alpha^{a_V})$ 来代替 $\mathrm{Sig}_V(\alpha^{a_U},\alpha^{a_V})$。不幸的是，$W$ 不能计算 V 对 $(\alpha^{a_U},\alpha^{a_V})$ 的签名，因为他不知道 V 的签名算法 Sig_V。类似地，W 也不能用 $\mathrm{Sig}_U(\alpha^{a_U},\alpha^{a_V})$ 来代替 $\mathrm{Sig}_U(\alpha^{a_U},\alpha^{a_V})$，因为他不知道 U 的签名算法 Sig_U。可见，正是签名的使用阻止了中间入侵攻击。

上述描述的简化了的 STS 协议没有提供密钥确证性。然而，它可被很容易地修改为具有这种性质的协议，只需做如下修改：在第(4)步，定义 $y_V=E_K(\mathrm{Sig}_V(\alpha^{a_U},\alpha^{a_V}))$；在第(6)步，定义 $y_U=E_K(\mathrm{Sig}_U(\alpha^{a_U},\alpha^{a_V}))$。修改后的协议称作端-端协议。

10.3.2 MTI 密钥协定协议

Matsumoto，Takashima 和 Imai 通过修改 Diffie-Hellman 密钥交换协议已经构造了一些有趣的密钥协定协议[9]。我们将这些协议称为 MTI 协议，这些协议不需要用户 U 和 V 之间计算任何签名。它们是两段协议(two-pass protocol)，而 STS 协议是一个三段协议(three-pass protocol)。这里我们只介绍 MTI 协议中的一个。假定 p 是一个素数，α 是 Z_p 的一个本原元，p 和 α 是公开知道的。每个用户 U 有一个 ID 串 $\mathrm{ID}(U)$，一个秘密指数 a_U($0\leqslant a_U\leqslant p-2$)和一个相应的公开值 $b_U=\alpha^{a_U}\bmod p$。可信中心有一个数字签名方案，将其公开的验证算法记为 Ver_{TA}，秘密的签名算法记为 Sig_{TA}。每个用户有一个证书 $C(U)=(\mathrm{ID}(U),b_U,\mathrm{Sig}_{TA}(\mathrm{ID}(U),b_U))$，这里 $b_U=\alpha^{a_U}\bmod p$。下面我们来描述 MTI 密钥协定协议：

(1) U 随机地选择一个数 r_U，$0\leqslant r_U\leqslant p-2$，并计算 $S_U=\alpha^{r_U}\bmod p$；

(2) U 发送 $(C(U),S_U)$ 给 V；

(3) V 随机地选择一个数 r_V，$0\leqslant r_V\leqslant p-2$，并计算 $S_V=\alpha^{r_V}\bmod p$；

(4) V 发送$(C(V),S_V)$给 U；

(5) U 从 $C(V)$中获得 b_V 并计算 $K=S_V^{a_U}b_V^{r_U}\bmod p$。$V$ 从 $C(U)$中获得 b_U 并计算 $K=S_U^{a_V}b_U^{r_V}\bmod p$。

在 MTI 协议中传输的信息可图示为

$$U \xrightarrow{C(U),\alpha^{r_U}\bmod p} V$$
$$U \xleftarrow{C(V),\alpha^{r_V}\bmod p} V$$

关于 MTI 协议的安全性论证见文献[9]。该协议中没有使用数字签名，似乎协议不能抵抗中间入侵攻击。的确，一个主动的敌手 W 改变 U 和 V 相互发送的值是可能的。这个过程可描述为

$$U \xrightarrow{C(U),\alpha^{r_U}\bmod p} W \xrightarrow{C(U),\alpha^{r_U}\bmod p} V$$
$$U \xleftarrow{C(V),\alpha^{r_V}\bmod p} W \xleftarrow{C(V),\alpha^{r_V}\bmod p} V$$

此时，U 计算 $K=\alpha^{r_U a_V+r_V a_U}\bmod p$，$V$ 计算 $K=\alpha^{r_U a_V+r_V a_U}\bmod p$。然而，因为 W 不知道 U 和 V 的秘密指数 a_U 和 a_V，所以他不能计算 U 和 V 的任何一个的密钥。这表明，即使 U 和 V 已经计算了不同的密钥，W 也不能计算这两个密钥中的任何一个。换句话说，U 和 V 确信对方是网络中能计算他们已经计算的密钥的唯一的一个用户。这个特性有时称为隐式密钥认证(implicit key authentication)。

10.3.3 Girault 密钥协定协议

Girault 密钥协定协议[10]不需要证书。用户的公开密钥的值和他自己的身份隐含着相互认证。它组合了 RSA 和离散对数的特点。假定 $n=pq$，$p=2p_1+1$，$q=2q_1+1$，p，q，p_1 和 q_1 均为大素数。设 α 是 Z_n^* 中阶为 $2p_1q_1$ 的元素。n 的分解只有可信中心知道。n 和 α 是公开的，p，q，p_1 和 q_1 都是秘密的。可信中心选择一个公开的 RSA 加密指数 e。保密对应的解密指数 $d=e^{-1}\bmod\varphi(n)$。每一个用户 U 有一个 ID 串 ID(U)。

用户 U 按下列步骤从可信中心获得一个自证明公钥(self-certifying public key)p_U：

(1) U 选择一个秘密指数 a_U，并计算 $b_U=\alpha^{a_U}\bmod n$；

(2) U 将 a_U 和 b_U 提供给可信中心；

(3) 可信中心计算 $p_U=(b_U-\mathrm{ID}(U))^d\bmod n$；

(4) 可信中心将 p_U 发送给 U。

现在我们来描述 Girault 密钥协定协议：

(1) U 随机地选择一个数 r_U，计算 $S_U=\alpha^{r_U}\bmod n$；

(2) U 将 ID(U)，p_U 和 S_U 发送给 V；

(3) V 随机地选择一个数 r_V，计算 $S_V=\alpha^{r_V}\bmod n$；

(4) V 将 ID(V)，p_V 和 S_V 发送给 U；

(5) U 计算 $K=S_V^{a_U}(p_V^{e}+ID(V))^{r_U}\bmod n$，$V$ 计算 $K=S_U^{a_V}(p_U^{e}+ID(U))^{r_V}\bmod n$。

在 Girault 密钥协定协议中传输的消息可图示为

$$U \xrightarrow{\mathrm{ID}(U),\ p_U,\ \alpha^{r_U} \bmod n} V$$
$$U \xleftarrow{\mathrm{ID}(V),\ p_V,\ \alpha^{r_V} \bmod n} V$$

在 Girault 密钥协定协议中，因为可信中心没有对值 $\mathrm{ID}(U)$，p_U 和 b_U 进行签名，所以任何别的人没有办法直接验证它们的真实性。假定这些信息由想伪装作 U 的敌手 W 伪造。如果 W 从 $\mathrm{ID}(U)$和一个伪造的值 b'_U 开始，那么他无法计算 b'_U 所对应的指数 a'_U。不知道 a'_U，W 就不能假装作 U 来计算一个密钥。像 MTI 协议中一样，可说明 Girault 密钥协定协议提供了隐式认证这一特性。

从 Girault 密钥协定协议我们可以看出，可信中心无需 U 给它提供 a_U，它就能直接从 b_U 计算出 p_U。现在要问为什么还要要求 U 把值 a_U 提供给可信中心呢？实际上，U 提交 a_U 可使可信中心在为 U 计算 p_U 之前相信 U 的确知道值 a_U。现在我们来说明如果可信中心不通过检测用户拥有的值 a_U 及对应的 b_U 就随意地给用户颁布公钥 p_U，那么 Girault 协议可被攻击。

假定 W 选择一个伪造的假 a'_U，并计算对应的值 $b'_U=\alpha^{a'_U} \bmod n$。现在 W 确定对应的公钥 $p'_U=(b'_U-\mathrm{ID}(U))^a \bmod n$。$W$ 计算 $b'_W=b'_U-\mathrm{ID}(U)+\mathrm{ID}(W)$，并将 b'_W 和 $\mathrm{ID}(W)$发送给可信中心。假定可信中心为 W 颁布公钥 $p'_W=(b'_W-\mathrm{ID}(W))^a \bmod n$，易知，$b'_W-\mathrm{ID}(W)\equiv(b'_U-\mathrm{ID}(U))(\bmod n)$，故 $p'_W=p'_U$。

假定 U 和 V 执行 Girault 协议，W 按下列方式代替信息：

$$U \xrightarrow{\mathrm{ID}(U),\ p_U,\ \alpha^{r_U} \bmod n} W \xrightarrow{\mathrm{ID}(U),\ p'_U,\ \alpha^{r_U} \bmod n} V$$
$$U \xleftarrow{\mathrm{ID}(V),\ p_V,\ \alpha^{r_V} \bmod n} W \xleftarrow{\mathrm{ID}(V),\ p_V,\ \alpha^{r_V} \bmod n} V$$

现在 V 计算密钥 $K'=\alpha^{r'_U a_V+r_V a'_U} \bmod n$，$U$ 计算密钥 $K=\alpha^{r'_U a_V+r_V a'_U} \bmod n$。$W$ 能计算 $K'=S_V^{a'_U}(p'_V+\mathrm{ID}(V))^{r'_U} \bmod n$。这样 W 和 V 共享一个密钥，但 V 认为他和 U 共享一个密钥。所以 W 能解密由 V 发送给 U 的消息。

关于密钥分配和协定的综述性文章可参阅文献[11]。关于基于身份的密钥分配方案可参阅文献[12]。

10.4 密钥的保护和秘密共享

10.4.1 密钥的保护

密钥的安全保密是密码系统安全的重要保证，保证密钥安全的基本原则是除了在有安全保证环境下进行密钥的产生、分配、装入以及存储于保密柜内备用外，密钥决不能以明文形式出现。

(1) 终端密钥的保护。可用二级通信密钥(终端主密钥)对会话密钥进行加密保护。终端主密钥存贮于主密钥寄存器中，并由主机对各终端主密钥进行管理。主机和终端之间就可用共享的终端主密钥保护会话密钥的安全。

(2) 主机密钥的保护。主机在密钥管理上担负着更繁重的任务，因而也是敌手攻击的

主要目标。在任一给定时间内，主机可有几个终端主密钥在工作，因而其密码装置须为各应用程序所共享。工作密钥存储器要由主机施以优先级别进行管理，并对未被密码装置调用的那些会话密钥加以保护。可采用一个主密钥对其它各种密钥进行加密，称此为主密钥原则。这种方法将对大量密钥的保护问题化为仅对单个密钥的保护。在有多台主机的网络系统中，为了安全起见，各主机应选用不同的主密钥。有的主机采用多个主密钥对不同类密钥进行保护，例如用主密钥 0 对会话密钥进行保护，用主密钥 1 对终端主密钥进行保护，而网中传送会话密钥时所用的加密密钥为主密钥 2。三个主密钥可存放于三个独立的存贮器中，通过相应的密码操作进行调用，作为工作密钥对其所保护的密钥加密、解密。这三个主密钥也可由存贮于密码器件中的种子密钥(seed key)按某种密码算法导出，以计算量来换取存贮量的减少。此法不如前一种方法安全。除了采用密码方法外，还必须和硬件、软件结合起来，以确保主机主密钥的安全。

(3) 密钥分级保护管理法。图 10.4.1 和表 10.4.1 给出了密钥分级结构。

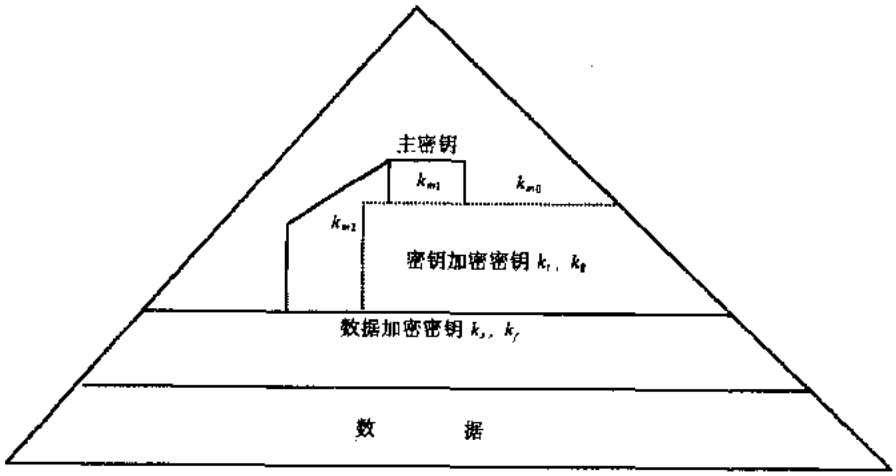


图 10.4.1　密钥分级保护

表 10.4.1　密钥分级结构

密钥种类	密钥名	用途	保护对象
密钥加密密钥	主机主密钥 0，k_{m0} 主机主密钥 1，k_{m1} 主机主密钥 2，k_{m2}	对现有密钥或存贮在主机内的密钥加密	初级密钥 二级密钥 二级密钥
	终端主密钥(或二级通信密钥)k_t 文件主密钥(或二级文件密钥)k_g	对主机外的密钥加密	初级通信密钥 初级文件密钥
数据加密密钥	会话(或初级)密钥 k_s 文件(或初级)密钥 k_f	对数据加密	传送的数据 存贮的数据

从上述结构可以看出，大量数据可以通过少量动态产生的数据加密密钥(初级密钥)进行保护，而数据加密密钥又可由更少量的、相对不变(使用期较长)的密钥加密(二级)密钥或主机主密钥 0 来保护，而其它主机主密钥(1 和 2)用来保护二级密钥。这样只有极少数密钥以明文形式存贮在有严密物理保护的主机密码器件中，其它密钥则以加密后的密

文形式存于密码器之外的存贮器中，因而大大地简化了密钥管理，并改善了密钥的安全性。为了保证密钥的安全，在密码设备中都有防窜扰装置，当密封的关键密码器件被撬开时，其基本密钥和主密钥等会自动从存贮器件中清除，或启动装置自动引爆。

10.4.2 秘密共享

存贮在系统中的所有密钥的安全性(从而整个系统的安全性)可能最终取决于一个主密钥。这样做有两个缺陷：一是若主密钥偶然地或蓄意地被暴露，整个系统就易受攻击；二是若主密钥丢失或毁坏，系统中的所有信息就用不成了。后一个问题可通过将密钥的副本发给信得过的用户来解决。但这样做时，系统对背叛行为又无法对付。解决这两个问题的一个办法是使用秘密共享方案(secret sharing scheme)。秘密共享方案的基本观点是：将密钥 k 按下述方式分成 n 个共享(share)$k_1, k_2, \ldots, k_n$：

(1) 已知任意 t 个 k_i 值易于算出 k；

(2) 已知任意 $t-1$ 个或更少个 k_i，则由于信息短缺而不能决定出 k。这种方法也称为 (t,n)门限(threshold)法[13]。

将 n 个共享 $k_1, k_2, \ldots, k_n$ 分给 n 个用户。由于要重构密钥要求至少有 t 个共享，故暴露 $s(s\leqslant t-1)$个共享不会危及密钥，且少于 t 个用户的组不可能共谋得到密钥。同时，若一个共享被丢失或毁坏，仍可恢复密钥(只要至少有 t 个有效的共享)。这种方法也可用于保护任何类型的数据。下面我们来介绍两个秘密共享方案。

10.4.2.1 Shamir 门限方案

Shamir[13]于 1979 年基于拉格朗日内插多项式提出了一个门限方案。本小节来介绍这一方案。

假定 p 是一个素数，共享的密钥 $k\in K=Z_p$。可信中心给 $n(n<p)$个共享者 $P_i(1\leqslant i\leqslant n)$分配共享的过程如下：

(1) 可信中心随机选择一个 $t-1$ 次多项式 $h(x)=a_{t-1}x^{t-1}+\ldots+a_1x+a_0\in Z_p[x]$，常数 $a_0=k$；

(2) 可信中心在 Z_p 中选择 n 个非零的、互不相同的元素 $x_1, x_2, \ldots, x_n$，计算 $y_i=h_i(x_i), 1\leqslant i\leqslant n$；

(3) 将 $(x_i, y_i)(1\leqslant i\leqslant n)$分配给共享者 $P_i(1\leqslant i\leqslant n)$，值 $x_i(1\leqslant i\leqslant n)$是公开知道的，$y_i(1\leqslant i\leqslant n)$作为 $P_i(1\leqslant i\leqslant n)$的秘密共享。

每对 (x_i, y_i)就是"曲线"$h(x)$上的一个点。因为 t 个点唯一地确定 $t-1$ 次多项式 $h(x)$，所以 k 可以从 t 个共享重构出。但是从 $t_1(t_1<t)$个共享无法确定 $h(x)$或 k。

给定 t 个共享 $y_{i_s}(1\leqslant s\leqslant t)$，从拉格朗日多项式重构的 $h(x)$为

$$h(x)=\sum_{s=1}^{t} y_{i_s}\prod_{\substack{j=1\\ j\neq s}}^{t}\frac{x-x_{i_j}}{x_{i_s}-x_{i_j}}$$

运算都是 Z_p 上的运算。

一旦知道 $h(x)$，通过 $k=h(0)$易于计算出密钥 k。因为

$$k=h(0)=\sum_{s=1}^{t} y_{i_s}\prod_{\substack{j=1\\ j\neq s}}^{t}\frac{-x_{i_j}}{x_{i_s}-x_{i_j}}$$

若令

$$b_s = \prod_{\substack{j=1 \\ j \neq s}}^{t} \frac{-x_{i_j}}{x_{i_s} - x_{i_j}}$$

则

$$k = h(0) = \sum_{s=1}^{t} b_s y_{i_s}$$

因为 $x_i(1 \leqslant i \leqslant n)$的值是公开知道的，所以我们可预计算 $b_s(1 \leqslant s \leqslant n)$以加快重构时的运算速度。

10.4.2.2 Asmuth-Bloom 门限方案

Asmuth 和 Bloom[14]于 1980 年基于中国剩余定理提出了一个门限方案。在他们的方案中，共享是和密钥 k 相联系的一个数的同余类。令 $p, d_1, \ldots, d_n$ 是满足下列条件的一组整数：

(1) $p > k$；

(2) $d_1 < d_2 < \cdots < d_n$；

(3)对所有的 i，$\gcd(p, d_i) = 1$；对 $i \neq j$，$\gcd(d_i, d_j) = 1$；

(4) $d_1 d_2 \cdots d_t > p d_{n-t+2} d_{n-t+3} \cdots d_n$。

令 $n = d_1 d_2 \cdots d_t$ 是 t 个最小的 d_i 之积，则 n/p 大于任意 $t-1$ 个 d_i 之积。令 r 是区域 $[0, n/p-1]$中的一个随机整数。为了将 k 划分成 n 个共享，计算 $k' = k + rp$，$k' \in [0, n-1]$。n 个共享为

$$k_i = k' \bmod d_i \qquad i = 1, 2, \cdots, n$$

为了恢复 k，找到 k'就足够了。若给定 t 个共享 $k_{i_1}, \cdots, k_{i_t}$，则由中国剩余定理可知，同余方程组

$$x' \equiv k_{i_1} (\bmod\ d_{i_1})$$
$$x' \equiv k_{i_2} (\bmod\ d_{i_2})$$
$$\cdots$$
$$x' \equiv k_{i_t} (\bmod\ d_{i_t})$$

关于模 $n_1 = d_{i_1} d_{i_2} \cdots d_{i_t}$在 $[0, n_1 - 1]$内有唯一解 x，因为 $n_1 \geqslant n$，这就唯一地确定了 k'，即 $k' = x \bmod n$。最后，从 k'，r 和 p 计算 k：$k = k' - rp$，即 $k = k' \bmod p$。

若仅知道 $t-1$ 个共享 $k_{i_1}, k_{i_2}, \cdots, k_{i_{t-1}}$，可能就只知道 k'关于模 $n_2 = d_{i_1} d_{i_2} \cdots d_{i_{t-1}}$在 $[0, n_2 - 1]$内有唯一解 x。因为 $n/n_2 > p$，$\gcd(p, n_2) = 1$，所以使 $x \leqslant n$ 和 $x \equiv k'$的数 x 在模 p 的所有同余类上均匀地分布，因此，没有足够的信息去决定 k'。关于该方案的详细讨论和进一步修正请参阅文献[14]。

目前已利用各种方法诸如代数方法、组合方法和几何方法构造出了大量的适应于各种不同环境的秘密共享方案，诸如有欺骗者参加的秘密共享、多重秘密共享。关于构造方面，可进一步参阅文献[15,18,19,20,22,26,27,28,29,30,31,32,33,35,36]等。关于秘密共享方案的有效性等性质的研究以及与其它方面的关系，可参阅文献[16,17,21,23]。另外，文献[8,24,25,37]对秘密共享方案也作了很好的阐述。

10.5 密钥托管技术

1993年4月16日，美国政府宣布了一项新的建议，该建议倡导联邦政府和工业界使用新的具有密钥托管功能的联邦加密标准[38]。该建议称为托管加密标准(escrowed encryption standard(EES))，又称Clipper建议。其目的是为用户提供更好的安全通信方式，同时允许政府为了国家安全监听某些通信。这个建议的核心是一个新的称之为Clipper的防窜扰芯片，它是由NSA主持开发的硬件实现的密码设备。采用了称之为Skipjack的私钥密码算法，芯片的单元密钥(UK)由两个称之为Escrow的机构联合提供。1994年2月，美国政府公布采用EES[39]。下面我们来详细描述这一标准。

10.5.1 Clipper芯片的构成

Clipper芯片采用Skipjack算法，它的密钥长度为80比特，明文和密文长度均为64比特，轮数为32，可采用ECB，CBC，OFB和CFB等四种工作模式。算法不公开，以便保护Escrow系统的安全。该算法于1985年由NSA开始设计，并于1990年完成评价工作，为机密级算法(目前该算法已公开)。设计者们声称该算法有很强的安全性(事实上，该算法的安全强度很低)。我们用E表示Skipjack算法，用$E(K,M)$表示用一个80比特长的会话密钥K加密明文M所得的密文。

每个芯片都有一个唯一的芯片识别符(ID)，一个80比特长的单元密钥(UK)和一个80比特长的族密钥(FK)。各芯片的单元密钥互不相同，但所有芯片的族密钥均相同。我们用chip_i表示用户i的clipper芯片，其唯一的芯片识别符用ID_i表示。

因为Clipper芯片是防窜扰的(tamper-resistant)，所以没有人(包括芯片的拥有者)能访问它的内容。因此，Skipjack算法可得到很好的保护。

10.5.2 Clipper芯片的编程

所有芯片均由安全密封信息设备(SCIF-Secure Compartmented Information Facility)进行编程，由两个Escrow协助完成。SCIF中有一台袖珍计算机和对芯片写入程序的设备。在一次会话中可对300个芯片进行编程。将芯片识别符ID、单元密钥UK、族密钥FK及特殊控制软件写入芯片中的VROM或VIA-Link存贮器中。SCIF存放在Mykotronx公司。编程过程如图10.5.1所示。

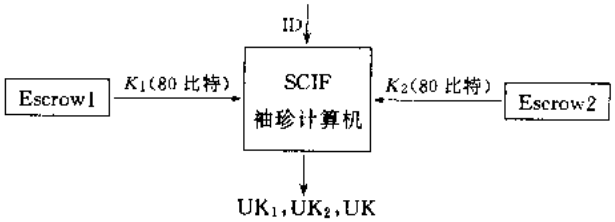


图10.5.1 芯片的编程过程

K_1和K_2分别是Escrow1和Escrow2独立产生的两个80比特长的秘密随机串，ID是芯片的识别符。UK_1，UK_2和UK分别是系统所产生的单元密钥的两个分量和单元密钥。运行过程如下：

(1)每次会话时，Escrow1和Escrow2分别向SCIF送入80比特长的秘密随机串K_1和K_2作为各芯片和单元密钥的种子。单元密钥UK是K_1和K_2的函数，每个Escrow只掌握其中的一半。

(2)芯片识别符 ID 一般取为 32 比特长，将其填充成 64 比特长，以 ID_1 表示。计算 $R_1=E(K_1,D(K_2,E(K_3,ID_1)))$。类似地可对同一 ID 填充不同的比特串可获得 64 比特长的串 ID_2 和 ID_3，相应地有 $R_2=E(K_1,D(K_2,E(K_3,ID_2)))$，$R_3=E(K_1,D(K_2,E(K_3,ID_3)))$。

(3)将 R_1，R_2，R_3 级联成 $64\times3=192$ 比特长的串，取前 80 比特作为 UK_1，次 80 比特作为 UK_2，其余的弃之不用。单元密钥 $UK=UK_1\oplus UK_2$。

对每个赋予识别符的芯片都产生相应的 UK_1，UK_2 和 UK，分别存入三张软盘中。第一张盘存入识别符 ID 及 UK_1，发给Escrow1保存；第二张存入识别符 ID 及 UK_2，发给Escrow2保存；各个 Escrow 只知道自己送出的 80 比特的种子密钥和 80 比特的组成单元密钥的分量。第三张盘存入 ID 及 UK，用于对相应芯片编程。编程后，所有有关信息将从 SCIF 中消失。Escrow 将带自己的软盘离去。为了确保没有关键信息留下，可将袖珍计算机捣毁。

10.5.3 LEAF 和 Clipper 芯片的加解密过程

法律强制访问域(law enforcement access field(LEAF))是 EES 的一个主要特征。每当使用一个 Clipper 芯片时，为了监视它的使用，必须产生一个 LEAF。

假定用户 A 想使用他的 Clipper 芯片 $Chip_A$ 给另一个用户 B 发送一个消息 M。A 将首先和 B 用任一密钥交换方法(如 Diffie-Hellman 密钥交换方法)商定一个 80 比特长的秘密密钥作为会话密钥 K。然后 A 将 M 和 K 送入 $Chip_A$，经计算给出密文 $C=E(K,M)$ 和 $LEAF(A,K)$。LEAF 结构中包括 32 比特长的识别符、对 80 比特长的会话密钥的加密结果和一个 16 比特长的校验和。对 Clipper 芯片 $Chip_A$，当一个会话密钥 K 被用来加解密时，它的 LEAF 具有下列形式：$LEAF(A,K)=E(FK,D(A,K))$，其中 $D(A,K)=(ID_A,E(UK_A,K),f(A,K,IV))$，$f(A,K,IV)$是由某一秘密函数 f 产生的一个 16 比特校验和，IV 是一个初始向量。A 将 $C=E(K,M)$和 $LEAF(A,K)$发送给接收者 B。

Diffie 称 LEAF 为在芯片上为政府机构所开的“后门”。政府有关机构为了法律和国家安全的需要，可以通过有关法规进入此“后门”，对 Clipper 加密的数据进行解密。

当 B 收到 $C=E(K,M)$和 $LEAF(A,K)$时，他用他的芯片 $Chip_B$ 和会话密钥 K 对 $C=E(K,M)$进行解密。因为 Skipjack 算法是一个保密算法，所以即使 B 知道 K 和 $C=E(K,M)$，不使用 $Chip_B$ 也不能解密 $E(K,M)$。

用户 B 利用 $Chip_B$ 的解密过程如下：

(1)用户 B 将会话密钥 K，密文 C 和 $LEAF(A,K)$送入 $Chip_B$；

(2)$Chip_B$ 使用 FK 解密 $LEAF(A,K)=E(FK,D(A,K))$得 $D(A,K)=(ID_A,E(UK_A,K),f(A,K,IV))$；

(3)$Chip_B$ 计算 $f(A,K,IV)$并和收到的校验和比较，如果不匹配，停止，否则，进行第(4)步；

(4)$Chip_B$ 使用会话密钥 K 解密 $E(K,M)$获得明文 M。

由上述可知，在一个 Clipper 芯片内的校验和的验证对 EES 的应用是关键的。因为 EES 被封装在一个防窜扰硬件之中，没有人能跳过芯片内预先拟定好的某些步骤使用一个 Clipper 芯片。特别地，校验和验证不能被绕过。只有在这个验证成功地通过之后，解密操作才能被执行。换句话说，如果 A 想让他的消息被接收并可由一个接收者解密，他必须

提供一个合法的 LEAF。

10.5.4 授权机构的监听

假定一个政府机构想监听用户 A 给用户 B 发送的消息，该机构首先必须获得法庭的许可。然后，他向两个密钥托管机构提交法庭给他的监听用户 A 的委托书(warrant)和用户 A 的芯片识别符 ID_A(因为政府机构可拿到族密钥 FK，所以它可以先使用 FK 对 LEAF(A,K)解密获得用户 A 的芯片识别符 ID_A)。在两个密钥托管机构验证了委托书后，他们给政府机构分别提供用户 A 的密钥分量 UK_1 和 UK_2，则 $UK_A=UK_1 \oplus UK_2$。此时政府机构使用 FK 和 UK_A 解密 LEAF(A,K)获得会话密钥 K，这样他就能解密 $E(K,M)$获得 M。

没有法庭的许可，任何人(除了两个密钥托管机构合伙)不能获得任何 Clipper 芯片的单元密钥，因此没有人能找到任何特定的通信双方的会话密钥。这样，所有用户的秘密性就得到了保证。

10.5.5 LEAF 反馈攻击

EES 出台以后受到民间的强烈反对，除了 EES 有侵犯公民的隐私权外，众多密码学者也纷纷指出了 EES 的弱点[40,41,42,46]。文献[42]在指出 EES 的缺陷的同时，还给出了一种新的密钥托管体制。另外，文献[44]和[45]也设计了一些别的类型的密钥托管体制。文献[51]是关于密钥托管技术的一篇很好的综述性文章。我们来介绍一种攻击 EES 的方法，即 LEAF 反馈(LEAF Feedback)攻击方法。这种攻击可逃避政府机构的监听。设通信双方 A 和 B 选择一个单向函数 $h(\cdot)$。K 是 A 和 B 的会话密钥，令 $K'=h(K)$。LEAF 反馈攻击过程可描述：

(1) A 和 B 分别以 K 和 K' 产生真实的 LEAF(A,K)和假的 LEAF(A,K')；

(2) A 在发送以 K 加密的密文 $C=E(K,M)$之前，发送 LEAF(A,K')而不是 LEAF(A,K)；

(3) 接收者 B 解密时，以 LEAF(A,K)代替 LEAF(A,K')通过芯片的检验，然后用 K 解密密文；

(4) 政府截获的是 LEAF(A,K')，他无法恢复出 K，从而用户 A 可逃避政府的监听。

因为芯片是防窜扰的，所以(2)、(3)中的替代过程在信道的两端进行。

10.6 注记和文献

密钥管理是一个重要而又刺手的问题。目前有关密钥管理技术的讨论已有不少。文献[49]的第 12 章和第 13 章比较详细地介绍了现有的一些密钥管理技术。

“密钥托管”、“密钥恢复”和“可信第三方”本质上是一回事，有关这方面的讨论请查阅有关最新资料，因为这些技术还很不成熟，又是争论的焦点。

关于秘密共享的讨论请参阅文献[25,37]。但这些文献中讨论的秘密共享都假定了可信中心的存在。对无可信中心的秘密共享感兴趣的读者请参阅文献[50]。

参 考 文 献

[1] Meyer, C. H. and Matyas, S. M., Cryptography. A New Dimension in Computer Data Security—— A Guide for the Design and Implementation of Secure Systems, John Wiley & Sons, 1982.

[2] Blom, R., An Optimal Class of Symmetric Key Generation Schemes, Advances in Cryptology-Eurocrypt'84, Springer-Verlag, 1985, pp. 335—338.

[3] Blundo, C., Desantis, A., Herzberg, A., Kutten, S., Vaccaro, V. and Yung, M., Perfectly-secure Key Distribution for Dynamic Conferences, Advances in Cryptology-Crypto'92, Springer-Verlag, 1993, pp. 471—486.

[4] Beimel, A. and Chor, B., Interaction in Key Distribution Schemes, Advances in Cryptology-Crypto'93, Springer-verlag, 1994, pp. 444—455.

[5] Diffie, W., Van Oorschot, P. C. and Wiener, M. J., Authentication and Authenticated Key Exchanges, Designs, Codes and Cryptography, 2(1992), pp. 107—125.

[6] Diffie, W. and Hellman, M. E., Multiuser Cryptographic Techniques, AFIPS Conference Proceedings, 45(1976), pp. 109—112.

[7] Diffie, W. and Hellman, M. E., New Directions in Cryptography, IEEE Transactions on Information Theory, 22(1976), pp. 644—654.

[8] Schneier, B., Applied Cryptography, Protocols, Algorithms and Source Code in C, John Wiley and Sons, 1993.

[9] Matsumoto, T., Takashima, Y. and Imai, H., On Seeking Smart Public-key Distribution Systems, Transactions of the IECE(Japan), 69(1986), pp. 99—106.

[10] Girault, M., Self-certified Public Key, Advances in Cryptology-Crypto'92, Springer Verlag, 1991, pp. 490—497.

[11] Van Tilburg, J., Secret Key exchange with Authentication, Computer Security and Industrial Cryptography, State of the Art and Evolution, ESAT Course, May, 1991, Springer-Verlag, 1993, pp. 71—86.

[12] Harn, L. and Yang, S. B., ID-based Cryptographic Schemes for User Identification, Digital Signature and Key Distribution, IEEE J. Select. Areas Commun, Vol. 71, pp. 757—760.

[13] Shamir, A., How to Share a Secret, Comm. ACM, Vol. 22(11), 1979, pp. 612—613.

[14] Asmuth, C. and Bloom, J., A Modular Approach to Key Safeguarding, IEEE Transactions on Information Theory, Vol. IT-30, May 1983, pp. 208—210.

[15] Blakley, G. R., Safeguarding Cryptographic Keys, Proc. NCC, Vol. 48, AFIPS Press, Montvale, N. J., pp. 313—317(1979).

[16] Blakley, G. R., One-time Pads Are Key Safeguarding Schemes, Not Cryptosystems, Proc. 1980 symp. On Security and Privacy, IEEE Computer Society, pp. 108—113(Apr. 1980).

[17] Blakley, G. R. and Swanson, J., Security Proofs for Information Protection Systems, In Proc. 1981 Symp. On Security and Privacy, IEEE Computer Society, pp. 75—88(Apr. 1981).

[18] ITO, M., Saito, A. and Nishizeki, T., Secret Sharing Scheme Realizing General Access Structure, Proceedings IEEE Globecom'87, 1987, pp. 99—102.

[19] Benaloh, J. and Leichter, J., Generalized Secret Sharing and Monotone Functions, Advances in Cryptology Crypto'98, Springer-Verlag, 1990, pp. 27—35.

[20] Brickell, E. F., Some Ideal Secret Sharing Schemes, Journal of Combinatorial Mathematics and Combinatorial Computing, 9(1989), pp. 105—113.

[21] Capocelli, R. M., De Santis, A, Gargano, L. and Vaccaro, V., On the Size of Shares for Secret Sharing Schemes, Journal of Cryptology, 6(1993), pp. 157—167.

[22] Blundo, C, De Santis, A., Stinson, D. R. and Vaccaro, V., Graph Decompositions and Secret Sharing Schemes, Advances in Cryptology-Eurocrypt'92, Springer-Verlag, 1993, pp. 1—24.

[23] Brickell, E. F. and Davenport, D. M., On the Classification of Ideal Secret Sharing Schemes, Journal of Cryptology, 4(1991), pp. 123—134.

[24] Simmons, G. J., An Introduction to Shared Secret and/or Shared Control Schemes and Their Application, In Contemporary Cryptology, The Science of Information Integrity, pp. 441—497, IEEE Press, 1992.

[25] Stinson, D. R., An Explication of Secret Sharing Schemes, Designs, Codes and Cryptography, 2(1992), pp. 357—390.

[26] Kothari, S. C., Generalized Linear Threshold Scheme, Advances in Cryptology-Crypto'84, Springer-Verlag, 1985, pp. 231—241.

[27] Karnin, E. D., Greene, J. W. and Hellman, M. E., On Sharing Secret Systems, IEEE Transactions On Information Theory, Vol. IT-29, 1983. pp. 35—41.

[28] Desmedt, Y. and Frankel, Y., Threshold Cryptosystem Advances in Cryptology Crypto'90, Springer Verlag, 1990, pp. 307—315.

[29] Desmedt, Y. and Frankel, Y., Shared Generation of Authentication and Signatures, Advances in Cryptology Crypto'91, Springer Verlag, 1992, pp. 457—469.

[30] Tompa, M. and Woll, H., How to Share a Secret with Cheaters, Journal of Cryptology, Vol. 1, No. 2, 1988, pp. 133—138.

[31] Lin, H. Y., and Harn, L., A Generalized Secret Sharing Scheme with Cheater Detection, Advances in Cryptology-Asiacrypt'91, Springer-Verlag, 1993, pp. 149—158.

[32] Brickell, E. F. and Stinson, D. R., The Detection of Cheaters in Threshold Schemes, Advances in Cryptology-Crypto'88, Springer-Verlag, 1990, pp. 564—577.

[33] Benaloh, J. C., Secret Sharing Homorphisms: Keeping Shares of a Secret, Advances in Cryptology Crypto'86, Springer-Verlag, 1987, pp. 251—260.

[34] Beutelspacher, A., How to Say "No", Advances in Cryptology-Eurocrypt'89, Springer Verlag, 1990, pp. 491—496.

[35] 冯登国，肖国镇，数字通信中密钥共享方案的一个新设计，首届全国数字通信学术会议论文集，1993年5月，西安，159—161页.

[36] He, J. and Dawson, E., Multistage Secret Sharing Based on One way Function, IEEE letters, Vol. 30, No. 19, 1994, pp. 1591—1592.

[37] Stinson, D. R., Cryptography-Theory and Practice, CRC Press, 1995.

[38] White House Press Release, The Clipper Chip Initiative, IEEE Information Society Newsletter, Vol. 43, No. 2, June 1993, pp. 21—23.

[39] NIST, Escrowed Encryption Standard, Federal Information Processing Standards Publication 185, U. S. Dept. of Commerce, 1994.

[40] Blaze, M., Protocol Failure in the Escrowed Encryption Standard, Proceedings of the 2nd ACM Conference on Computer and Communications Security, 1994, pp. 59—67.

[41] Frankel, Y. and Yung, M., Escrow Encryption Systems Visited: Attacks, Analysis and Designs, Advances in Cryptology-Crypto'95, Springer-Verlag, 1995, pp. 222—235.

[42] He, J. M. and Dawson, E., A New Key Escrow, Cryptosystem, Cryptography: Policy and Algorithms, pp. 105—114, 1995.

[43] Denning, D. E. and smid, M., Key Escrowing Now, IEEE Communications Magazine, Sep. 1994, pp. 54—68.

[44] Micali, S. and Sidney, R., A Simple Method for Generating and Sharing Pseudo-Random Functions, with Applications to Clipper-like Key Escrow Systems, Advances in Cryptology-Crypto'95, Springer-Verlag, 1995, pp. 185—196.

[45] Lenstra, A. K., Winkler, P. and Yacobi, Y., A Key Escrow System with Warrant Bounds, Advances in Cryptology-Crypto'95, Springer-Verlag, 1995, pp. 197—207.

[46] Kilian, J. and Leighton, T., Fair Cryptosystems Revisited, Advances in Cryptology-Crypto'95, Springer Verlag, 1995, pp. 208—221.

[47] Denning, D. E. R., Cryptography and Data Security, Addison-Wesley Publishing Company, 1982.

[48] 王育民、何大可，保密学——基础与应用，西安电子科技大学出版社，西安，1990.

[49] Menezes, A. J., Van Oorschot, P. C. and Vanstone, S. A., Handbook of Applied Cryptography, CRC Press, 1996.

[50] Ingemarsson, I., Simmons, G. J., A Protocol to Set up Shared Secret Schemes Without the Assistance of a Mutually Trusted Party, Advances in Cryptology-Crypto'90, Springer-Verlag, 1991, pp. 266—282.

[51] Denning, D. E., A Taxonomy for Key Escrow Encryption Systems, Communications of the ACM, 39(3), March, 1996.

第 11 章　电子货币及其它

11.1　电子货币的分类及其特点

货币是从商品世界中分离出来的，固定地充当一般等价物的特殊商品。它是商品交换的必然产物，是商品内在矛盾发展的必然结果。商品流通是货币流通的基础，货币流通是商品流通的反映。作为流通手段的货币，最初都是金属条、金属块，每次商品交换时都用称来称份量和查成色，很不方便。随着商品经济的发展、市场的扩大，就逐渐出现由国家铸造的货币即铸币。经过不断的实践和探索，人们意识到可以用其它材料制造的货币符号来代替金属货币以充当流通手段的职能。这样，像征性的货币或价值符号——纸币就出现了。纸币是指由国家发行并强制使用的价值符号，它代替金属货币充当流通手段的职能。

随着社会的信息化和电子化以及远距离贸易的增多，纸币也面临着严峻的挑战，此时使用的货币必须能在网上进行传输，而纸币显然已不适应这种发展。因而 80 年代初人们就开始从理论上研究电子货币，特别是近几年兴起了一股电子货币热。目前有些国家和地区正在试行。在信息高速公路的建设中，金融系统是一个重要的国家信息基础设施，而电子货币是金融系统中的重要研究内容之一。因而研究电子货币无论是在理论上还是在应用上都有着重要的意义。目前，研究电子货币的基本出发点有两个。一个是不考虑个人隐私权，也就是说银行能追踪顾客的每一笔开支，顾客不能隐瞒把钱交给了谁，购买了什么东西。这种电子货币称为可追踪的电子货币，目前，可追踪的电子货币是很容易实现的，利用现有的密码技术如加密技术和认证技术便能设计出满足要求的可跟踪的电子货币。银行目前所关注和实施的都是这种电子货币。但从发展的角度来看，这种电子货币不会得到顾客的支持，因为这种电子货币不能提供保护个人隐私的能力。为此，人们从另一角度出发来研究电子货币，即考虑个人的隐私权，使用这种电子货币银行不能追踪顾客的开支情况，不能知道顾客把钱给了谁，购买了什么东西。这种电子货币称为不可追踪的电子货币。与可追踪的相比，这种电子货币的设计并非是一件容易的事情，目前主要还停留在理论研究上，即使理论也很不成熟，诸如系统的设计太复杂，一些关键技术还未彻底解决。

由于设计可追踪的电子货币容易、简单，所以本章就不再赘述。本章主要来介绍不可追踪的电子货币。

现实中的货币可用皮夹来存放，用支票来支取，但电子货币要用智能卡作为电子钱夹来存贮，存取都是通过智能卡来实现的。现用的货币的安全性主要依赖于其物理特性，诸如伪造、复制钞票和硬币的困难性。而电子货币的安全性不依赖于任何物理条件，它的安全性必须从数学上来保证，密码学技术本质上是用来保证电子货币的安全性的。信息本身有一个价值，电子货币能通过网络来传输。

电子货币按支付方式可分为在线电子货币和离线电子货币两种。在线电子货币要求每次支付都要有银行的参加。离线电子货币在支付过程中无需和银行联系。在线电子货币主要阻止超额消费，它的通信代价很高，一般适用于高额支付，对低额支付是不实用的。

离线电子货币主要阻止资金的滥用，一般适用于低额支付，而对高额支付是不适用的。

电子货币按面值是否变化可分为两种：一种是硬币(coin)，面值固定不变；另一种是支票(check)，面值在不断变化。

一个电子货币至少应具有以下三个特点：

(1) 独立性。电子货币的安全性不能依赖于任何物理条件，这样货币能通过网络传输。

(2) 安全性。能阻止伪造和拷贝货币，不可重复使用。

(3) 不可追踪性。用户的秘密性能得到保护，也就是说，用户和他的购买之间的关系对任何人来说都是不可追踪的。

一个理想的电子货币应和现实中的货币具有相同的特点，除了具有上述三个特点外还应具有以下三个特点：

(1) 可迁移性。货币能迁移给别的用户，这也说明能将货币借给别人。

(2) 可分性。能把价值为 C 的货币分割成许多子片，每个子片具有任何期望的值，并且其值不大于 C，但这些子片的总价值必须等于 C。

(3) 离线支付。用户在支付过程中无需和银行取得联系。

文献[9]中设计了一个理想的电子货币系统，但太复杂，不实用，即使后来进行了改进，仍很复杂，不实用。目前所设计的系统除了必须满足性质(1)～(3)外，一般满足性质(4)～(6)中的部分性质。

11.2 在线电子货币

一个电子货币系统主要由三个协议组成：提取协议(用户从银行提取货币的协议)，支付协议(用户向商店付款的协议)，存款协议(用户向银行存款的协议)。电子货币的不可追踪性主要由盲数字签名来保证。关于在线电子货币，已有许多种实现方案。本章我们主要来介绍基于 7.6 节中的 RSA 盲数字签名方案设计的几个在线电子货币方案。

方案一

银行 B 选择两个大素数 p 和 q，以及一个单向函数 f，令 $n=pq$，并选择 e 使得 e 和 $\varphi(n)=(p-1)(q-1)$ 互素，由 $ed\equiv 1 \bmod \varphi(n)$ 求出 d。现在银行 B 公开 n,e,f，秘密保存 p,q，及 d。银行 B 首先制定货币的单位，对任何消息 m，约定 $f(m)$ 的 e 次根，即 d 次幂值 1 元人民币。在这里我们使用在线支付方式，即每次交易都必须和银行联系，商店介于用户和银行之间。

首先，用户 A 从银行 B 取钱：A 随机选择 $m,r\in Z_n^*$，计算 $m_1=f(m)r^e$，并将 A 的身份 ID_A，帐号 N_A 及 m_1 发送给 B。银行 B 验证 A 的身份证 ID_A，若合法，银行 B 就对 m 进行签名，即计算 $m_1^{1/e}=m_1^d$，并从 A 的帐号 N_A 中去掉 1 元人民币。这里假定 A 已在银行 B 存了一笔钱。B 将 $m_1^{1/e}=m_1^d$ 发送给 A，用户 A 计算 $m_1^{1/e}/r=(f(m)r^e)^d/r=f^d(m)$。

用户 A 得到了银行 B 对 m 的真实签名 $f^d(m)$。$f^d(m)$ 值 1 元人民币，就是用户 A 从银行取出的货币。在这里使用盲数字签名技术，实现了用户 A 的不可追踪性。

其次，用户 A 从商店 S 买 1 元钱的东西，并付给商店 1 元人民币。一般来说，还要涉及找零钱，因而需要找零钱协议，该方案不考虑这个问题。由于是在线支付，在用户 A 付

款的过程中，银行 B 把商店 S 的 1 元人民币同时也存入 S 在 B 的帐号之中，即存储过程也同时得到了完成。用户 A 将$(m, f^d(m))$发送给商店 S，S 将$(ID_S, N_S, m, f^d(m))$发送给 B，B 验证收到的签名，并检查 m 是否在以前被花过。如果签名合法并且 m 没有被花过，那么就在 S 的帐号 N_S 上存入 1 元人民币。

方案二

银行首先制定货币的单位。银行选择两个大素数 p, q，以及一个单向函数 f，公开 $n=pq$ 和 f，保密 p 和 q。选择一系列奇素数 $e_0, e_1, \cdots, e_{t-1}$，并将 $e_0, e_1, \cdots, e_{t-1}$ 公开，对任何消息 m，约定 $f(m)$ 的 $e_i(0 \leqslant i \leqslant t-1)$ 次根即 $f(m)^{1/e_i}$ 值 2^t 元人民币。对于一个给定的支付量 $x(0 \leqslant x \leqslant 2^t-1)$，设 $x=x_0+x_1 2+x_2 2^2+\cdots+x_{t-1}2^{t-1}$，令

$$e(x)=\prod_{\substack{x_i=1\\0\leqslant i\leqslant t-1}} e_i$$

则 $f(m)$的 $e(x)$ 次根值 x 元人民币。最大的支付量为 2^t-1 元人民币，例如 $f(m)$的 e_0e_1 次根值 3 元人民币。

对于消息 m 的一个签名(比如 $f(m)^{1/e_ie_j}$)可通过某些幂次(比如 e_j)得出另一个签名(比如$(f(m)^{1/e_ie_j})^{e_j}=f(m)^{1/e_i}$)，这表明，一个硬币可通过公开的指数进行降值。例如，条子 $f(m)^{1/e_0e_1}$值 3 元人民币，则对 $f(m)^{1/e_0e_1}$进行 e_1 次幂后得 $f(m)^{1/e_0}$，此时，条子 $f(m)^{1/e_0}$值 1 元人民币。

现在我们来描述一个电子货币方案，该方案允许用户(支付者)将花剩的零钱积累起来。可以在下次提取钱时存入他的银行帐号上。在这个方案中，用户(支付者)周期地从银行提取一定量的条子，每张条子都值 2^t-1 元人民币(2^t-1 是系统所能提供的条子的最大值)。假定现在用户 A 要从银行提取两张条子。提取过程如下：用户 A 随机选择 n_i，$r_i \in Z_n^*$，并计算

$$f(n_i)r_i^h(i=1,2), h=\prod_{i=0}^{t-1} e_i$$

A 将 $f(n_i)r_i^h(i=1,2)$，ID_A 和 N_A 发送给 B，银行 B 检查 A 的身份 ID_A，如果合法，B 对 $f(n_i)r_i^h(i=1,2)$签名，即计算$(f(n_i)r_i^h)^{1/h}(i=1,2)$并在 A 的帐号 N_A 上去掉 $2(2^t-1)$元人民币。这里假定 A 已在他的银行帐号 N_A 上存了一笔钱。B 将 $f^{1/h}(n_i)r_i(i=1,2)$发送给 A，A 计算

$$f^{1/h}(n_i)r_i/r_i=f^{1/h}(n_i)(i=1,2)$$

用户 A 现在想从商店 S 买值 3 元人民币的东西，但他提取的每张条子都值 2^t-1 元人民币，他需要将零钱积累起来，他的第一次支付过程如下：用户 A 计算$(f(n_1)^{1/h})^{e_2\cdots e_{t-1}}=f(n_2)^{1/e_0e_1}$。随机选择 $j, s_1 \in Z_N^*$，计算 $f(j)s_1^{e_2e_3\cdots e_{t-1}}$，并将 n_1，$f(n_1)^{1/e_0e_1}$和 $f(j)s_1^{e_2e_3\cdots e_{t-1}}$ 发送给 S。S 将 ID_S，N_S，n_1，$f(n_1)^{1/e_0e_1}$和 $f(j)s_1^{e_2e_3\cdots e_{t-1}}$ 发送给 B。B 验证收到的签名，并检查 n_1 以前是否被花过，如果签名合法并且 n_1 以前未被花过，那么就在 S 的帐号 N_S 上记入 3 元人民币，并计算

$$(f(j)s_1^{e_2e_3\cdots e_{t-1}})^{1/e_2e_3\cdots e_{t-1}}$$

将$(f(j))^{1/e_2e_3\cdots e_{t-1}}s_1$ 发送给 S。然后，S 将$(f(j))^{1/e_2e_3\cdots e_{t-1}}s_1$ 发送给 A。A 计算

$$(f(j))^{1/e_2e_3\cdots e_{t-1}}s_1/s_1=(f(j))^{1/e_2e_3\cdots e_{t-1}}$$

并验证$(f(j))^{1/e_2e_3\cdots e_{t-1}}$。

A 用 $f(j)^{1/e_2\cdots e_{t-1}}$来积累他所花剩的零钱，这里使用了另一个 RSA 模 N，N 的分解只有银行 B 知道，N 专用于积累零钱的运算。

如果用户 A 又想从商店 S 买值 1 元人民币的东西，他的第二次支付过程如下：用户 A 计算$(f(n_2)^{1/h})^{e_1\cdots e_{t-1}}=f(n_2)^{1/e_0}$。随机选择 $s_2\in Z_N^*$，计算$(f(j))^{1/e_2e_3\cdots e_{t-1}}s_2^{e_1e_2\cdots e_{t-1}}$，并将

$$n_2, f(n_2)^{1/e_0} \text{ 和 } (f(j))^{1/e_2e_3\cdots e_{t-1}}s_2^{e_1e_2\cdots e_{t-1}}$$

发送给 S。S 将 $ID_S, N_S, n_2, f(n_2)^{1/e_0}$和$(f(j))^{1/e_2e_3\cdots e_{t-1}}s_2^{e_1e_2\cdots e_{t-1}}$ 发送给 B。B 按 S 申报的价值验证收到的签名，并检查 n_2 以前是否被花过，如果签名合法并且 n_2 以前未被花过，那么就在 S 的帐号 N_S 上记入 1 元人民币，并计算

$$((f(j))^{1/e_2e_3\cdots e_{t-1}}s_2^{e_1e_2\cdots e_{t-1}})^{1/e_1e_2\cdots e_{t-1}}$$

将$((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}s_2$ 发送给 S。然后，S 将$((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}s_2$ 发送给 A。A 计算

$$((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}s_2/s_2 = ((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}$$

并验证$((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}$。

用户 A 用同一个 $f(j)$来积累他花剩的零钱，在提取下一批条子时，用户 A 将积累起来的钱存入他的帐号，存贮过程如下：A 将 j 和$((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}$发送给银行 B，B 按 A 提供的 $e_i(0\leqslant i\leqslant t-1)$的重数来验证$((f(j))^{1/e_2e_3\cdots e_{t-1})^{1/e_1e_2\cdots e_{t-1}}}$，并检查 j 是否以前被存储过，如果签名合法并且 j 未被存储过，则 B 在 A 的帐号上记入 $2(2^t-1)-4$ 元人民币，并存储 j。

方案三

货币单位的制定与方案二中的相同，所不同的只是方案二中将所花剩的零钱积累了起来，而该方案不是将花剩的零钱积累起来，而是将这些零钱分配到没有花过的条子上，并且后来他们自己能花这些钱。方案二中为了积累这些零钱使用了另一个 RSA 模 N，该方案只使用了一个 RSA 模 n。用户 A 从银行 B 提取钱(或条子)的过程与方案二中的一样。设 d 是用户 A 支付的面额所对应的指数，g 是条子的值所对应的指数，$c=g/d$ 是花剩的钱所对应的指数，$h=e_0e_1\cdots e_{t-1}$是系统中最大的面额所对应的指数，这里 $d|g|h$。用户 A 的支付过程如下：

用户 A 随机选择 $s\in Z_n^*$，计算 ms^c，m 具有某种特定的形式，它可能是在 f 之下的某些像的乘积，下面我们将要解释。A 将 $n, c, f(n)^{1/d}, ms^c$ 发送给 S。然后 S 将 $ID_S, N_S, n, c, f(n)^{1/d}, ms^c$ 发送给 B，B 验证签名，并检查 n 是否以前被花过，如果签名合法，并且以前未被花过，则 B 在 S 的帐号 N_S 上记入 d 元人民币，并计算 $m^{1/c}sf(f(n)^{1/c})$，$f(f(n)^{1/c})$是一个保护因子，以确保用户 A 必有 $f(n)^{1/c}$。B 将 $m^{1/c}sf(f(n)^{1/c})$发送给 S，然后 S 将 $m^{1/c}sf(f(n)^{1/c})$发送给 A。A 计算

$$f(n)^{1/c} = (f(n)^{1/g})^d \text{ 和 } m^{1/c}sf(f(n)^{1/c})/sf(f(n)^{1/c}) = m^{1/c}.$$

并验证之。

下面我们来描述用户 A 分配零钱到没有使用过的条子上的过程：

不妨假设 $d=e_0e_1$，$g=e_0e_1e_2e_3$，则 $c=g/d=e_2e_3$，取 m 为 $f^{e_2}(n_1)f^{e_3}(n_2)$即 $m\equiv f^{e_2}(n_1)f^{e_3}(n_2)$。$n_1$ 和 n_2 是 A 随机选择的，但 n_1 和 n_2 他从未使用过。这样，用户 A 执行完支付协议后得到 $a\equiv m^{1/e_2e_3}$。

用户 A 可通过 a 计算出 $f(n_1)^{1/e_3}$和 $f(n_2)^{1/e_2}$，即可得到两张条子，计算过程如下：

$$u \equiv e_2^{-1} \bmod e_3$$

$$v \equiv e_2 u \text{ div } e_3 (v \text{ 是 } e_2 u \text{ 除以 } e_3 \text{ 所得的商})$$

$$f(n_1)^{1/e_3} \equiv (a^{e_2} f(n_2)^{-1})^u f(n_1)^{-v}$$

$$f(n_2)^{1/e_2} \equiv a\, f(n_1)^{-1/e_3}$$

这样就将剩余的钱分配到了两张条子上。

11.3 离线电子货币

离线电子货币的设计要比在线电子货币的设计复杂得多。在离线电子货币的设计中，要解决的最关键的问题是多重花费问题。一个用户总可以在不同的商店里花同样的硬币，这是因为支付是离线的而且用户又是匿名的，事后无法检测出伪造者。在现有的离线电子货币系统中，一部分系统的设计采用了“切割-选择”(cut and choose)技术，这类系统不仅复杂，而且效率低。另一部分系统的设计未采用“切割-选择”技术，而是使用了一些特殊的技巧，这类系统不仅简单，而且有效。

本节我们介绍一个比较简单而有效的离线电子货币系统，该系统的安全性是基于素数阶群的表示问题。

设 G_q 是一个阶为素数 q 的群，$k \geqslant 2$，$g_i \in G_q \backslash \{1\}$，$g_i \neq g_j (i \neq j)$，如果对任何 $h \in G_q$，存在 $a_i \in Z_q$，$1 \leqslant i \leqslant k$，使得 $h = \prod_{i=1}^{k} g_i^{a_i}$，那么我们称 $(g_1, g_2, \cdots, g_k)$ 是 G_q 的一个长度为 k 的生成重(generator-tuple)，$(a_1, a_2, \cdots, a_k)$ 称为 h 关于 $(g_1, g_2, \cdots, g_k)$ 的一个表示。

素数阶群的表示问题可描述为：给定一个群 G_q 和它的一个生成重 $(g_1, g_2, \cdots, g_k)$，以及 $h \in G_q$，求 h 关于 $(g_1, g_2, \cdots, g_k)$ 的一个表示。为方便起见而又不失一般性，我们设 G_q 是 Z_p^* 的一个阶为 q 的乘法子群，其中 p 和 q 均为素数，$q \mid (p-1)$。

现在我们来描述离线电子货币系统。用 **B** 表示银行，用 S 表示商店，用 U 表示用户(顾客)。一般来说，在实际中 U 是一个支付器件(诸如 Smart 卡、个人计算机等)，但我们这里将 U 等同于用户(顾客)。

系统的建立：银行 **B** 随机产生一个生成重 (g, g_1, g_2) 和一个数 $x \in Z_q^*$，同时选择两个具有如下形式的自由碰撞的单向 Hash 函数 H 和 H_0：

$$H : G_q^5 \to Z_q^*$$

$$H_0 : G_q^2 \times \text{SHOP} - \text{ID} \times \text{DATE/TIME} \to Z_q$$

函数 H 用于银行的签名的产生和验证，而函数 H_0 用于支付协议中确定口令。**B** 公开 G_q 的描述(即公开 p 和 q)，生成重 (g, g_1, g_2) 以及 H 和 H_0 的描述作为它的公钥，**B** 的秘密密钥是 x。

每一个商店 S 有一个唯一的识别号 I_S(I_S 可能是 S 在银行 **B** 的帐号)，**B** 和 S 都知道 I_S，记所有这样的识别号的全体为 SHOP-ID。将 SHOP-ID 作为 H_0 的输入，可确保两个不同的商店以很大的概率产生不同的口令。DATE/TIME 表示交易的日期/时间之集，将 DATE/TIME 作为 H_0 的输入，可确保同一个商店在每次支付中产生不同的口令。

B 关于一个对 $(A, B) \in G_q^2$ 的一个签名 Sign(A, B) 是一个满足下列条件的四重组 $(z, a, b, r) \in G_q^4$：

$$g^r = h^{H(A,B,z,a,b)}a$$
$$A^r = z^{H(A,B,z,a,b)}b$$

其中 $h=g^x$。

一个硬币(比如值1元人民币)是一个三重组$(A,B,\mathrm{Sign}(A,B))$。如果U知道A和B关于(g_1,g_2)的一个表示,那么我们就说U知道硬币的一个表示。

银行为顾客开帐号的过程:银行**B**有一个存储帐号的数据库。当顾客U在银行**B**开帐号时,**B**要求U证明其身份(比如出示身份证或护照等)。U随机产生一个数$u_1\in Z_q^*$,计算$I=g_1^{u_1}$,如果$g_1^{u_1}g_2\neq 1$,那么U将I发送给**B**,并秘密保存u_1。**B**在帐号数据库中存储U的识别信息和I。我们称I为U的帐号。帐号的唯一性是很必要的,因为在多重花费的情况下,它能使**B**唯一地识别U。同时**B**计算$z=(Ig_2)^x$,并将z发送给U。

提取(款)协议:当U想从他的银行帐号上提取一个硬币时,他首先必须证明他是该帐号的合法拥有者,比如U使用数字签名对一个提款请求签名或用别的办法来识别他自己。然后,执行下列的提取协议:

(1)**B**随机产生一个数$w\in Z_q$,计算$a=g^w$和$b=(Ig_2)^w$,并将a和b发送给U。

(2)U随机产生三个数$s\in Z_q^*$,$x_1,x_2\in Z_q$,计算$A=(Ig_2)^s$,$z'=z^s$和$B=g_1^{x_1}g_2^{x_2}$。U也同时随机产生两个数$u,v\in Z_q$,计算$a'=a^ug^v$,$b'=b^{su}A^v$,$c'=H(A,B,z',a',b')$,$c=c'/u \bmod q$。然后,他将盲口令c发送给**B**。

(3)**B**给U发送回一个回答$r=(cx+w)\bmod q$,并从U的帐号上去掉一个硬币。U接收,当且仅当$g^r=ah^c$且$(Ig_2)^r=bz^c$。如果这个验证成立,U计算$r'=(ru+v)\bmod q$,则$\mathrm{Sign}(A,B)=(z',a',b',r')$是$(A,B)$的一个合法签名。$(A,B,\mathrm{Sign}(A,B))$就是一个硬币,只有U知道该硬币的一个表示。

从提取协议我们可以看出,**B**从未看到过(z',a',b',r'),本质上,该协议是一个盲签名协议,因此,U可匿名地花钱。

支付协议:当U想向商店S支付硬币时,他和S执行下列协议:

(1)U将$A,B,\mathrm{Sign}(A,B)$发送给S。

(2)如果$A\neq 1$,那么S计算口令$d=H_0(A,B,I_S,\mathrm{date/time})$,并将其发送给U。其中$I_S$表示S的唯一识别号,date/time是指U和S的交易日期/时间。

(3)U计算回答$r_1=(du_1s+x_1)\bmod q$和$r_2=(ds+x_2)\bmod q$,并将r_1和r_2发送给S。

如果$\mathrm{Sign}(A,B)=(z',a',b',r')$是$(A,B)$的一个合法签名,并且$g_1^{r_1}g_2^{r_2}=A^dB$,那么S接收该硬币。

从支付协议可以看出,只有知道一个硬币的一个表示的用户才能花一个硬币。

存储(款)协议:在一段时间之后(因为系统是离线的),S将支付记录和交易的日期/时间发送给**B**,支付记录由$A,B,\mathrm{Sign}(A,B),r_1$和$r_2$构成。如果$A=1$,那么**B**不接收支付记录。否则,**B**首先使用S的识别号I_S,S提供的支付记录和交易日期/时间计算d。然后,**B**验证$g_1^{r_1}g_2^{r_2}=A^dB$是否成立,$\mathrm{Sign}(A,B)=(z',a',b',r')$是否是$(A,B)$的一个合法签名。如果二者至少有一个不成立,那么**B**拒绝接收S的支付记录。否则,**B**搜索它的存储数据库看A是否曾被存储过。此时,有两种可能性:一种是A没有被存储过,在这种情况下,**B**在它的存储数据库中以S的名义存储$(\mathrm{A},\mathrm{date/time},r_1,r_2)$,并在S的帐号上记入一个硬币;另一种是A已被存储在存储数据库之中,在这种情况下,一定出现了伪造,要么是S

试图存储第二次，此时，已存的记录、日期/时间和新提交的记录、日期/时间完全一样，要么是U重花了该硬币，此时，口令是不同的。因为现在**B**在它的数据库中从新的记录中得到一对(d,r_1,r_2)，从已存储的信息中得到一对(d',r_1',r_2')，所以有$g_1^{r_1}g_2^{r_2}=A^dB$和$g_1^{r_1'}g_2^{r_2'}=A^{d'}B$，从而$g_1^{r_1-r_1'}g_2^{r_2-r_2'}=(Ig_2)^{sd-sd'}$，又$sd-sd'=r_2-r_2'$，故$g_1^{r_1-r_1'}=I^{r_2-r_2'}$，即$I=g_1^{(r_1-r_1')/(r_2-r_2')}$，这说明**B**能计算$g_1^{(r_1-r_1')/(r_2-r_2')}$。此时**B**从它的帐号数据库中搜索这个帐号。对应的帐号拥有者就是双重花费者。数$(r_1-r_1')/(r_2-r_2')\bmod q$用作双重花费者的一个证明，它等于$\log_{g_1}I$，$I$是双重花费者的帐号。

上述讨论表明，该方案可阻止多重花费。

11.4 电子货币和完全犯罪

不可追踪的电子货币在保护用户的个人隐私的同时，也给犯罪分子带来了可乘之机。现在假定银行使用的是11.2节中的方案一。犯罪分子A可进行如下的完全犯罪活动：

A的犯罪过程如下[15]：

(1)A绑架一个婴儿；

(2)A选择t个消息$m_1,m_2,\cdots,m_t$(t大到他想要的那么多)；

(3)A盲变换这t个消息，他把这t个盲消息送给当局并威胁当局做下列两件事情，否则，他杀死婴儿：

(a)让银行签所有的t个盲消息；

(b)在报纸上公布结果；

(4)当局答应A的要求；

(5)A买一张报纸，恢复那些消息，并开始花它们，当局没有办法追踪这些签名来抓住他；

(6)A释放那个婴儿。

因为整个犯罪过程不需要接触，也没有带任何实际特征的标记，所以警察很难有机会抓住绑架者。

由此可见，电子货币系统也不是完美无缺的，它与现金系统一样，也存在着一些漏洞和不足，还需要继续研究和完善它。

11.5 电子选举协议

在民主社会中，选举总是使用一个锁着的箱子，该箱子的顶部有一个可放入选票的小缝，每次选举的选票是由选举委员会特制的。在选举时，人们将填好的选票一个接一个地投入选票箱。这样做的目的是确保选票的匿名性。随着计算机技术和通讯技术的发展，信息处理和分配的量和速度都迅猛增加。以前的处理方式显然已经跟不上技术的发展，电子选举已成为可能。

从理论上来讲，一个选举方案是具有秘密输入值的一个多成员的计算[29,30]，它使得输出的正确性是可检验的。

从实际应用来讲，一个选举方案可通过加密、盲签名和别的密码技术来实现。

一个理想的电子选举协议应满足以下一些特性：

(1)完全性(completeness)。所有合法选票都能被正确地统计(计数)。

(2)合理性(soundness)。不诚实的选举者不能扰乱选举。

(3)秘密性(privacy)。所有的选票都是秘密的，即能保护个人隐私。

(4)不可重复性(unresuability)。任何选举者都不能投两次票。

(5)合法性(eligibility)。只有经许可的选举者才能投票。

(6)公平性(fairness)。任何事情都不能影响选举。

(7)可验证性(verifiability)。所有选举者都能检验他们的选票在最后的表中是否被统计上。

目前人们已经提出了许多选举协议，在这些协议中，有的满足上述全部特性，有的满足上述部分特性，有的除满足上述全部或部分特性外还满足别的一些特性。它们中有的使用了盲签名技术，有的未使用这种技术。限于篇幅，我们不能在这里做一一介绍，感兴趣的读者可参阅文献[31,32,33,34,35,36, 37,38,39,40]。本节我们主要来介绍一个使用盲签名技术设计的满足上述全部特性的选举协议，称为 FOO 选举协议[40]。

在 FOO 选举协议中，主要有三种参加者，即选举者，一个选举管理中心和一个计票者(计票者可能由一个公开的广告牌来代替)。该协议的设计需要三种密码技术，即比特承诺(Bit-commitment)技术、普通的数字签名技术和盲数字签名技术。后两种技术，前面已经讨论过，但比特承诺技术还没有介绍过，在介绍 FOO 选举协议以前，我们先来介绍比特承诺技术。

11.5.1 比特承诺

一般地，一个比特承诺方案就是一个函数 $f:\{0,1\}\times X\to Y$，这里 X 和 Y 是两个有限集。$b\in\{0,1\}$的一个加密是$\{f(b,x)\}x\in X$ 中的某一个值。要求一个比特承诺方案满足以下两个特性：

(1)隐蔽性(Concealing)。对 $b\in\{0,1\}$，接收者不能从 $f(b,x)$确定出 b 的值。

(2)约束性(Binding)。发送者能“打开”$f(b,x)$，即发送者能通过揭露用于加密 b 的 x 的值使接收者相信 b 的确是被加密的值。发送者不能将 $f(b,x)$既“打开”成 0，又“打开”成 1。

如果发送者想承诺任何比特串 s，那么他可以通过独立地承诺 s 的每一个比特来完成。此时，仍记为 $f(s,k)$。

实现比特承诺方案的方法很多[31,41,42]，这里仅介绍两种实现方法。

方法一：使用描述在 6.5 节中的 Goldwasser-Micali 概率加密体制来实现比特承诺方案。

设 $n=pq$，p 和 q 都是素数，$m\in\overline{QR}(n)=\{x|(x/p)=(x/q)=-1\}$，公开 n 和 m，n 的分解只有发送者知道。设 $X=Y=Z_n^*$。$f(b,x)=m^bx^2\bmod n$。发送者通过选择一个随机数 x 加密 b，加密结果为 $y=f(b,x)$。当发送者想“打开”y 时，它揭露值 b 和 x，接收者验证 $y=m^bx^2(\bmod n)$。如果假定二次剩余问题是不可行的，那么 $f(b,x)$没有泄露关于 b 和 x 的任何信息。所以该方案满足隐蔽性。现在我们来说明该方案满足约束性。若该方案不

满足约束性，则存在 $x_1, x_2 \in Z_n^*$，使得 $mx_1^2 \equiv x_2^2 \bmod n$，这样 $m \equiv (x_2 x_1^{-1})^2 \bmod n$。说明 $m \in QR(n)$，即 m 是一个二次剩余，这与 $m \in \overline{QR}(n)$ 相矛盾。故上述方案也满足约束性。

方案二：基于离散对数问题来实现比特承诺方案。

由 4.4 节中的关于离散对数的比特安全性的讨论可知，当 $p \equiv 3 \pmod 4$ 是一个使得在 Z_p^* 上计算离散对数问题是不可行的素数时，一个离散对数的第二个低位比特(SGL)是安全的。这是我们将要介绍的比特承诺方案的安全性基础。

设 $X=\{1,2,3,\cdots,p-1\}$，$Y=Z_p^*$，用 SLB(x)表示整数 x 的第二个低位比特，则

$$\mathrm{SLB}(x)=\begin{cases}0 & x \equiv 0,1 \bmod 4\\ 1 & x \equiv 2,3 \bmod 4\end{cases}$$

比特承诺方案 f 定义为

$$f(b,x)=\begin{cases}\alpha^x \bmod p & \mathrm{SLB}(x)=b\\ \alpha^{p-x} \bmod p & \mathrm{SLB}(x)\neq b\end{cases}$$

由于 $p \equiv 3 \pmod 4$，所以可证明 $\mathrm{SLB}(p-x) \neq \mathrm{SLB}(x)$。

假定 Z_p^* 上的离散对数问题是不可行的，则可说明上述方案满足隐蔽性和约束性。

11.5.2 FOO 选举协议

在 FOO 选举协议中，选举管理中心所使用的盲数字签名方案可以是任何一种盲数字签名方案，为叙述方便，我们这里选择 7.6 节中介绍的基于 RSA 的盲数字签名方案。

预备阶段：选举者 V_i 选择并填写一张选票 v_i。随机选择一个密钥 k_i，用比特承诺方案 f 加密 v_i，即计算 $x_i=f(v_i,k_i)$。随机选择一个盲因子 $r_i \in Z_n^*$ 盲化 x_i，即计算 $e_i=r_i^e H(x_i) \bmod n$，并对 e_i 签名得 $s_i=\sigma_i(e_i)$。然后将(ID_i,e_i,s_i)发送给选举管理中心 A。其中 σ_i 表示 V_i 的签名方案，ID_i 为 V_i 的身份证，H 是一个公开的单向函数。

颁发选举证书阶段：选举管理中心 A 检查选举者 V_i 有无权力选举，如果 V_i 无权参加选举，则 A 拒绝给 V_i 颁发选举证书；否则，A 检查 V_i 是否已经申请过一个选举证书，如果 V_i 已申请过，A 拒绝再给 V_i 颁发。否则，A 检查 s_i 是否是消息 e_i 的合法签名，如果是，则 A 对 e_i 签名，即计算 $d_i=e_i^d \bmod n$ 并将 d_i 发送给 V_i，作为 A 颁布给 V_i 的选举证书。在颁发选举证书阶段末，A 宣布已获得选举证书的选举者的总数并公布包含有(ID_i,e_i,s_i)的一张表。

投票阶段：选举者 V_i 通过对 d_i 脱盲恢复 x_i 的签名 $y_i=d_i/v_i \bmod n$。V_i 检查 y_i 是否是 A 对 x_i 的合法签名，如果不是，V_i 通过向 A 证明(x_i,y_i)的不合法性并选用另一个 v_i 来重新获取选举证书，否则，V_i 匿名地将(x_i,y_i)发送给计票者 C。

收集选票阶段：计票者 C 通过使用 A 的签名验证方程检查 y_i 是否是 x_i 的合法签名。

表 11.5.1　收集选票阶段的选票表

序号	选票和附加信息
1	x_j,y_j
⋮	⋮
l	x_i,y_i
⋮	⋮

表 11.5.2　统计选票阶段的选票表

序号	选票和附加信息
1	x_j,y_j,k_j,v_j
⋮	⋮
l	x_i,y_i,k_i,v_i
⋮	⋮

如果是，C 将(l,x_i,y_i)填入表中(见表 11.5.1)，其中 l 是(x_i,y_i)的编号。在所有的选举者投完票后，C 公开该表。

公开阶段：选举者 V_i 检查选票的数量是否等于选票者的数量。如果不相等，要求选举者公开这些缺少的票在加密时使用的 r_i。V_i 检查他的选票是否列在表中，如果没有列在表中，他公开(x_i,y_i)、合法的选票和它的签名并要求列入表中。V_i 用序号 l 将密钥 k_i 即(l,k_i)匿名地发送给 C。

统计选票阶段：计票者 C“打开”选票 x_i 的承诺，恢复选票 v_i 并检查 v_i 是否是合法的选票。C 统计(计数)选票并宣布选举结果(见表 11.5.2)。

关于 FOO 选举协议的安全性论证见文献[40]。

本节最后，我们来谈谈电子货币和电子选举之间的关系。原则上，几乎所有已知的电子货币系统都能被用来构作选举协议。现在我们以 11.3 节中介绍的离线电子货币系统为例来说明具体的构作过程。

让银行对应选举管理中心，设 $C=(A,B,Z',a',b',r')$是选举者从银行提取的一个硬币，选举者不是花硬币 C，而是使用 C 来进行选举。此时选举者自己选择口令 d，而不是由商店来选择，将选举者的口令 d 之集来作为选举者的选票集，选票是(C,r_1,r_2,d)。计票者检查是否 C 是一个合法的硬币和等式 $g_1^{r_1}g_2^{r_2}=A^dB$ 是否成立；如果两者都成立，则计票者计数选举者的票。这种选举协议的安全性由电子货币的安全性来保证。

11.6 潜 信 道

G. Simmons 于 1983 年首次提出了潜信道(subliminal channel)的概念[43]，它的基本思想是通信双方通过交换完全无关的、签了名的消息来传送秘密消息，使得第三方即使看到他们的所有通信也无法知道寓于通信中的秘密信息。一个简单的潜信道可以是句子中单词的个数。句子中有奇数个单词对应“1”，而偶数个单词对应“0”。因此，当你读这种仿佛无关的段落时，我已将 0,1 串消息发送给了在场的我方的人。当然，这种算法不安全。设计安全性好的潜信道方案是可能的，现在已有一些具体方案。潜信道签名方案与通常的签名方案没有什么不同。在潜信道方案中，第三方不仅不能读懂潜消息，而且他也不知道存不存在潜信道。潜信道的一个明显的应用是在间谍网中。如果每人都收发签名的消息，间谍在签名消息时发送潜消息就不会被注意到。

下面我们介绍一个基于 ElGamal 签名方案的潜信道签名方案[44]。

密钥的产生类似于基本的 ElGamal 签名方案，首先选择一个素数 p，两个随机数 g 和 r，使得 g 和 r 都小于 p，然后计算 $K=g^r \bmod p$。公开 K，g 和 p，保密 r。该通信双方分别为 A 和 B，他们都知道 r，这个密钥一方面用来发送和阅读潜消息，另一方面还用来对消息签名。

为使用消息 M'来发送潜消息 M，要求 $\gcd(M,p)=1$，$\gcd(M',p)=1$，$\gcd(M,p-1)=1$。发方 A 计算 $x=g^M \bmod p$，并由方程 $M'=rx+My \pmod{p-1}$解出 y，像在基本的 ElGamal 签名方案中一样，M'的签名是(x,y)。所有的人都能验证方程 $K^xx^y=g^{M'} \bmod p$ 来验证签名的合法性。如果 B 收到的(x,y)是 M'的一个合法签名，那么他可以恢复潜消息 M，恢复公式为 $M=y^{-1}(M'-rx) \pmod{p-1}$。

由上述讨论可知，ElGamal 数字签名方案可嵌入潜信道。人们自然会想到 DSS 是否可嵌入潜信道。G. Simmons 于 1993 年发现了 DSS 中的一种潜信道[46]。他认为，DSS 提供了至今发现的最适宜于潜通信的环境。NIST 和 NSA 还未对此潜信道作出评论，没有人知道他们是否早就知道潜信道。因为潜信道使不诚实的 DSS 实现者可以在每次签名时泄露密钥的一部分，从而对签名方案构成威胁。使用者在选用别人给你实现的 DSS 时要十分慎重。关于 DSS 的潜信道的详细讨论见文献[46]。

文献[43]和[45]还介绍了在签名方案中嵌入潜信道的其它方案。文献[47]还讨论了在 Fiat-Shamir 和 Feige-Fiat-Shamir 协议中嵌入潜信道的协议以及对其可能的滥用。事实上，任意签名方案都能转化成潜信道。

11.7 智力扑克协议

智力扑克类似于普通扑克，只是没有牌也没有口头通信，参加者之间的所有交换都必须用消息完成。任一牌手都有可能想骗对方。

公平比赛至少要满足以下要求：

(1)比赛必须从“公平发牌”开始。假定牌手们通过交换一系列消息实现了这一要求，则：(a)牌手应知道自己手中的牌，但不知道其他人的；(b)手中的牌应不相连贯；(c)所有可能的手中牌对每位牌手应是等可能的。

(2)在比赛中，牌手可能要从剩下的牌中补抓几张牌，这也要像(1)中所述那样公平地处理。牌手还必须能将牌出示给对方而不泄露他们的其余的牌的秘密。

(3)比赛结束时，牌手们应能检验比赛是否公平，以及他们的对手有没有骗人，特别是对赢家是否作弊感兴趣。

为叙述方便起见，我们假定只有两个牌手 A 和 B 玩牌，每个人都有一个秘密密钥，这些密钥在比赛结束以后才公开。设 E_A 和 D_A 分别表示 A 的加密变换和解密变换。E_B 和 D_B 分别表示 B 的加密变换和解密变换。加密变换必须是可换的，即对于任何消息 M：$E_A(E_B(m))=E_B(E_A(M))$。

以消息 $M_i(1\leqslant i\leqslant 52)$表示 52 张牌。B 开始发牌，发牌协议如下：

(1) B 对 52 个消息 $M_i(1\leqslant i\leqslant 52)$加密，得到 52 个加密的消息：$E_B(M_i)(1\leqslant i\leqslant 52)$，而后他随机地洗这一副加了密的牌并将其送给 A。

(2) A 随机地选出 5 个加密消息，并送回给 B，B 将这 5 个消息加密后留在手中。

(3) A 再随机地选出 5 个加密消息 $C_i(1\leqslant i\leqslant 5)$，他用自己的密钥对这 5 个消息加密，得到 $C_i'=E_A(C_i)(1\leqslant i\leqslant 5)$，而后，将这 5 个通过两次加密的消息发送给 B。

(4) B 用他的密钥对每个消息 $C_i'(1\leqslant i\leqslant 5)$进行解密，得到对于某个消息 M_j 的 $E_A(M_j)$：$D_B(C_i')=D_B(E_A(E_B(M_j)))=D_B(E_B(E_A(M_j)))=E_A(M_j)$。他将这 5 个消息送给 A。而后 A 用他的密钥将它们解密并留在手中。

在比赛过程中，可重复上述过程来添牌。

因为在模算术下指数是可换的，所以可使用 RSA 公钥算法来实现智力扑克协议。但用 RSA 算法来实现智力扑克协议会有少量信息被泄漏[35,48]。这主要是因为指数变换 $f(x)=x^e \bmod n$将二次剩余仍变为二次剩余的缘故。这个特性可被用来标记某些牌，比如

所有的“A”。一种解决方法是把所有的数都变成二次剩余，这是可能的，因为在 Z_p^*（p 为素数）上一个非二次剩余乘以一个非二次剩余就是一个二次剩余。文献[49]给出了另一种解决方法，但由于其太复杂而不实用。关于智力扑克协议的讨论，感兴趣的读者还可参阅文献[50,51,52]。

11.8 健忘传输协议

健忘传输(oblivious transfer)协议是由 Rabin 于 1981 年首次提出的一种协议。在这种协议中，A 可以概率 50%向 B 传送秘密。因此，B 有 50%的机会收到秘密和 50%的机会收不到秘密。另一方面，B 将知道他是否已收到秘密，而 A 则不知道 B 是否收到了秘密。这种协议可描述为：

(1) A 将两个奇素数 p 和 q 之积送给 B，p 和 q 表示他的秘密，例如它们可能是 RSA 解密变换的秘密参数。

(2) B 随机地选取一个数 x，$0<x<n$ 且 $\gcd(x,n)=1$，计算 $a=x^2 \bmod n$ 并将 a 送给 A。

(3) A 知道 p 和 q，计算 a 的四个根 $x,n-x,y,n-y$。他随机地取出一个根送给 B。

(4) 如果 B 收到 y 或 $n-y$，则他就可以从 x 和 y 计算出 p 或 q，计算公式为 $\gcd(x+y,n)=p$ 或 q。如果 B 收到 x 或 $n-x$，则他就什么也没得到。

因为 n 有两个不同的素因子，所以方程 $a=x^2 \bmod n$ 有四个根，这四个根可由中国剩余定理求得。

文献[16,54]还讨论了一些别的类型的健忘传输协议，感兴趣的读者可参阅之。

健忘传输协议可用于通过电话掷硬币和交换秘密等[53]。我们在这里只介绍经由电话掷硬币的协议。其问题是设计一种方案，B 可以点叫“正面”或“反面”，A 可以一种方式掷硬币，使每个人都有 50%的机会获胜。经由电话抛掷一个实际硬币显然不能满足条件，因为如果 B 点正面，A 就会说“对不起，反面”。Blum[53]给出的掷硬币(coin flipping)协议如下：

(1) A 选出两个大素数 p 和 q，并将 $n=pq$ 送给 B。

(2) B 检验 n 是否为素数、素数幂或偶数，如果是，A 就是骗人并认输，否则 B 取一 x，并将 $a=x^2 \bmod n$ 送给 A。

(3) A 计算 a 的四个根，随机地取出一个送给 B。

(4) 若 B 可分解 n，他就获胜。

也有一些别的掷硬币协议，感兴趣的读者参阅文献[31,55,56]。

11.9 注记和文献

从考虑个人隐私的观点出发，电子货币应具有保护个人隐私的特点。人们将这种电子货币称为不可追踪的电子货币。不可追踪的电子货币分为两种：一种是在线电子货币，这种货币的设计相对比较容易，但它适合于大额支付，不适合于小额支付，因为所花的通信代价太高；另一种是离线电子货币，这种货币的设计相对比较难，因为它要求电子货币在

满足不可追踪性的同时，要能阻止重复花费。有关在线电子货币的讨论请参考文献[4,6,60,63]等。有关离线电子货币的讨论请参考文献[5,8,9,10,11,12,13,14]等。因为电子货币是目前密码学研究中的一个热点问题，建议感兴趣的读者随时注意国内外的发展动向和最新研究成果。

有关电子选择协议、潜信道、健忘传输协议、智力扑克协议的进一步参考文献在本章介绍这些协议的过程中已作过交待。

关于秘密的部分或全部泄露、多成员计算协议、公正的密码体制的讨论可参考文献[31]及该文献中所提供的参考文献。

关于证书机构的讨论请参阅文献[7,36]及x.509。

关于图像和语音加密的讨论请参阅文献[64,66]。

关于信息系统安全的讨论请参阅文献[65,66,67]。

关于计算机网络安全的讨论请参阅文献[68,69]。

另外，对量子密码、混沌密码、神经密码感兴趣的读者请参阅文献[31,70,71]。关于量子密码的最新成果可在crypto'96论文集中找到。自1989年开始，Cryptologia杂志刊登了一系列有关混沌密码的论文。国内一些学者对此也比较感兴趣，发表了一些见解，主要刊登在《密码与信息》杂志上。关于神经密码讨论的甚少，目前还处在试验阶段，是否可用于设计和分析密码体制，还有待于进一步研究。作者认为神经密码的前景并不乐观。

最近，国际上一些学者热衷于研究隐形技术，对该研究方向感兴趣的读者请参阅文献[72]。

参 考 文 献

[1] Diffie, W., Hellman, M. E., New Directions in Cryptology, IEEE Transactions on Information Theory, v. IT-22, n. 6, Nov 1977, pp. 74—84.

[2] Rivest, R., Shamir, A., Adleman, L. M., A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Comunications of the ACM, v. 21, n. 2, Feb 1978, pp. 120—126.

[3] ElGamal, T., A Public-key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, Advances in Cryptology-CRYPTO'84, Springer-Verlag, 1985, 10—18.

[4] Chaum, D., Security Without Identification: Transaction Systems to Make Big Brother Obsolete, Communications of the ACM, Vl. 28, No. 10, 1985, pp. 1030—1044.

[5] Chaum, D., Fiat, A., Naor, M., Untraceable Electronic Cash, Advances in Cryptology-Crypto'88, Springer-Verlag, 1990, pp. 319—327.

[6] Simon, D., Anonymous Communication and Anonymous Cash, Advances in Cryptology-Crypto'96, Springer-Verlag, pp. 61—73.

[7] Chaum, D., Evertse, J. H., A Secure and Privacy Protecting Protocol for Transmitting Personal Information Between Organizations, Advances in Cryptology-Crypto'86, Springer-Verlag, 1986, pp. 118—167.

[8] Okamoto, T., Ohta, K., Disposable Zero-knowledge Authentications and Their Applications to Untraceable Electronic Cash, Advances in Cryptology-CRYPTO'89, Springer-Verlag, 1990, pp. 481—496.

[9] Okamoto, T., Ohta, K., Universal Electronic Cash, Advances in Cryptology-CRYPTO'91, Springer-Verlag, 1992, pp. 324—337.

[10] Chaum, D., Pedersen, T. P., Wallet Databases with Observers, Advances in Cryptology-CRYPTO'92, Springer-Verlag, 1993, pp. 89—105.

[11] Ferguson, N. , Single Term Off-line Coins, Advances in Cryptology-EUROCRYPT'93, Springer-Verlag, 1994, pp. 318—328.

[12] Brands, S. , Untraceable Off-line Cash in Wallet with Observers, Advances in Cryptology-CRYPTO'93, Springer-Verlag, 1994, pp. 302—318.

[13] Eng, T. , Okamoto, T. , Single-term Divisible Electronic Coins, Advances in Cryptology-EUROCRYPT'94, Springer-Verlag, 1995, pp. 306—319.

[14] Yacobi, Y. , Efficient Electronic Money, Advances in Cryptology-ASIACRYPT'94, Springer-Verlag, 1995, pp. 153—163.

[15] von Solms S. H. , and Naccache, D. , On Blind Signatures and Perfect Crimes, Computers and Security, Vol. 11, No. 6, 1992, pp. 581—583.

[16] Stadler, M. , Piveteau J. M. , and Camenisch, J. , Fair Blind Signatures, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, 1995, pp. 209—219.

[17] Hayes, B. Anonymous One-time Signatures and Flexible Untraceable Electronic Cash, Advances in Cryptology-Auscrypt'90, Springer-Verlag, 1990, pp. 294—305.

[18] Chaum, D. , Blind Signatures for Untraceable Payments, Advances in Cryptology-CRYPTO'82, Plenum Press, 1983, pp. 199—203.

[19] Chaum, D. , Blind Signatur System, Advances in Cryptology-CRYPTO'83, Plenum Press, 1984, p. 153.

[20] Guillou, L. C. , Ugon, M. , Smart Card-a highly Reliable and Portable Security Device, Advances in Cryptology-CRYPTO'86, Springer-Verlag, 1987, pp. 464—479.

[21] Damgard, I. B. , Payment Systems and Credential Mechanisms with Provable Security Against Abuse by Individuals, Advances in Cryptology-Crypto'88, Springer Verlag, 1990, pp. 328—335.

[22] Pfitzmann, B. , Waidner, M. , How to Break and Repair a "Provably Secure" Untraceable Payment System, Advances in Cryptology-Crypto'91, Springer-Verlag, 1992, pp. 338—350.

[23] Hirschfeld, R. , Making Electronic Refunds Safer, Advances in Cryptology-Crypto'92, Springer-Verlag, 1993, pp. 106—112.

[24] Ferguson, N. , Extensions of Single-term Coins, Advances in Cryptology-Crypto'93, Springer-Verlag, 1994, pp. 292—301.

[25] Brands, S. , Restrictive Blinding of Secret-key Certificates, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, 1995, pp. 231—247.

[26] Jakobsson, M. , Ripping Coins a Fair Exchange, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, 1995, pp. 220—230.

[27] Pfitzmann, B. , Schunter, M. , Waidner, How to Break Another "Provably Secure" Payment System, Advances in Cryptology-Eurocrypt'95, Springer-Verlag, 1995, pp. 121—132.

[28] Abe, M. , Fujisaki, E. , How to Date Blind Signatures, Advances in Cryptology-Asiacrypt'96, Springer-Verlag, 1996, pp. 244—251.

[29] Chaum, D. , Crepeau, C. and Damgard, I. , Multiparty Unconditionally Secure Protocols, In Proceedings of the Twentieth Annual ACM Symposium On Theory of Computing, pp. 11—19, 1988.

[30] Chaum, D. , Damgard, I. , van J. , de Graaf, Multiparty Computitions Ensuring Privacy of Each Party's Input and Correctness of the Results, Advances in Cryptology-Crypto'87, Springer-Verlag, 1988, pp. 87—119.

[31] Schneier, B. , Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons, New York, 2nd Edition, 1996.

[32] Salomaa, A. , Public-key Cryptography, Berlin: Springer-Verlag, 1990.

[33] Nurmi, H. , Salomaa, A. , Santean, L. , Secret Ballot Elections in Computer Networks, Computer&Security, V. 10, 1991, pp. 553—560.

[34] DeMillo, R. , Lynch, N. , Merritt, M. , Cryptographic Protocals, Proceedings of the 14 th Annual Symposium on the Theory of Computing, 1982, pp. 383—400.

[35] DeMillo,R. ,Merritt,M. , Protocals for Data Security,Computer,V. 16,N. 2,Feb 1983,pp. 39—50.

[36] Chen,L. D. ,Witness Hiding Proofs and Applications, Ph. d. diss. ,Aarhus University,DAIMI PB-477, August 1994.

[37] Cohen,J. D. ,Fischer,M. H. , A Robust and Verifiable Cryptographically Secure Election Scheme,Proceedings of the 26 th Annual IEEE Symposium on the Foundations of Computer Science,1985,pp. 372—382.

[38] Chaum,D. ,Elections with Unconditionally Secret Ballots and Disruptions Equivalent to Breaking RSA, Advances in Cryptology-Eurocrypt'88, Springer-Verlag, 1988, pp. 177—181.

[39] Iversen,K. R. ,A Cryptographic Scheme for Computerized General Elections, Advances in Cryptology-Crypto'91, Springer-Verlag, 1992, pp. 405—419.

[40] Fujioka,A. ,Okamoto,T. ,Ohta,K. ,A Practical Secret Voting Scheme for Large Scale Elections,Advances in Cryptology-Ausocrypt'92, Springer-Verlag, 1993,pp. 244—251.

[41] Naor,M. ,Bit Commitment Using Pseudo-randomness,Advances in Cryptology-Crypto'89, Springer-Verlag, 1990, pp. 128—136.

[42] Brassoud,G. ,Chaum,D. ,Crepeau,C. ,Minimum Disclosure Proofs of Knowledge,Journal of Computer and Systems Science,37(1988),pp. 156—189.

[43] Simmons,G. J. ,The Prisoner's Problem and the Subliminal Channel, Advances in Cryptology-Crypto'83, Plenum Press,1984,pp. 51—67.

[44] Simmons,G. J. , The Subliminal Channel and Digital Signatures, Advances in Cryptology-Eurocrypto'84, Springer-Verlag,1985,pp. 51—67.

[45] Simmons, G. J. , A Secure Subliminal Channel(?), Advances in Cryptology-Crypto'85, Springer-Verlag, 1986, pp. 33—41.

[46] Simmons, G. J. , Subliminal Communication is Easy Using the DSA, Advances in Cryptology-Eurocrypto'93, Springer-Verlag,1994,pp. 218—232.

[47] Desmedt,Y. ,Goutier,C. ,Bengio,S. ,Special Uses and Abuses of the Fiat-Shamir Passport Protocol, Advances in Cryptology-Crypto'87, Springer-Verlag,1988,pp. 21—29.

[48] Fortune,S. ,Merritt,M. ,Poker Protocols, Advances in Cryptology-Crypto'84, Springer-verlag, 1985, pp. 454—464.

[49] Goldwasser,S. ,Micali,S. ,Probabilistic Encryption and How to Play Mental Poker Keeping Secret all Partial Information, In Proceedings of the 14th ACM Symposium on the Theory of Computing,1982,pp. 279—299.

[50] Crepeau, C. ,A Secure Poker Protocol That Minimizes the Effect of Player Coalitions,Advances in Cryptology-Crypto'85, Springer-verlag,1986,pp. 73—86.

[51] Crepeau, C. ,A Zero-knowledge Poker Protocol That Achieves an Confidentiality of the Players' Strategy,or How to Achieve an Electronic Porker Face, Advances in Cryptology-Crypto'86, Springer-Verlag,1987,pp. 239—247.

[52] Yung,M. ,Cryptoprotocols:Subscriptions to a Public Key,the Secret Blocking,and the Muti-Player Mental Poker Game,Advances in Cryptology-Crypto'84, Springer-Verlag,1985,pp. 439—453.

[53] Blum,M. ,Three Applications of the Oblivious Transfer:1. Coin Flipping by Telephone,2. How to Exchange Secrets, 3. How to Send Certified Electronic Mail,Dept. EECS,Univ. of California,Berkeley,Calif. (1981).

[54] Bellare,M. ,Micali,S. , Non-interactive Oblivious Transfer and Applications, Advances in Cryptology-Crypto'89, Springer-Verlag,1990,pp. 547—557.

[55] Kilian, J. ,Uses of Randomness in Algorithms and Protocols, Cambridge:MIT Press,1990.

[56] Reyneri,J. M. ,Karnin,E. D. ,Coin Flipping by Telephone,IEEE Transactions on Information Theory,V. IT-30,n. 5,Sep. 1984,pp. 775—776.

[57] 冯登国,基于离散对数问题的盲数字签名方案,通信保密,No. 1,1997,31—34 页.

[58] 冯登国、戴英侠,电子货币——现代密码学应用的重要成就之一,数字通信,Vol. 24,No. 1,1997,29—30 页.

[59] 冯登国,一个离线电子支付系统的设计与分析,密码与信息,No. 2,1997,21—23 页.

[60] 冯登国、庄顺、戴英侠,一个在线电子支付系统的设计,97' 全国计算机信息保密学术年会, 1997 年 10 月,昆明,

135—138 页.
[61] 冯登国，一个公平的健忘传输协议的漏洞，密码与信息，No. 3，1997，17—19 页.
[62] 冯登国，一种新型的离线电子货币，通信保密，No. 4，1997，29—31 页.
[63] Chaum，D.，Online Cash Checks，Advances in Cryptology-Eurocrypto'89，Springer-Verlag，1990，pp. 288—293.
[64] Beker，H. J.，Piper，F. C.，Secure Speech Communications，Academic Press，1985.
[65] Denning，D. E. R.，Cryptergraphy and Data Security，Addison-Wesiey，1982.
[66] 王育民，何大可，保密学——基础与应用，西安电子科技大学出版社，西安，1990.
[67] Fisher，R. P.，Information System Security，Reading，MA，Prentice-Hall，1984.
[68] Bersom，T. A.，Beth，T. (eds)，Local Area Network Security，Workshop LANSEC'89，European Institute for System Security，Karlsruhe，FRG，April3—6，1989.
[69] Ford，W.，Computer Communications Security，PTR Prentice Hall，Englewood Cliffs，New Jersey，1994.
[70] Matthews，R. On the Derivation of a 'Chaotic' Encryption Algorithm，Cryptologia，13：29—42，1989.
[71] Lauria，F. E.，On Neurocryptology，Parallet Architecturs and Neural Networks，5：15—18，1990.
[72] Anderson，R. (ed.) Information Hiding，First International Workshop，Cambridge，U. K.，May/June，1996，Proceedings，Springer-Verlag.

附 录

1. 数 学 基 础

密码学是一门交叉学科，它涉及到许多学科，诸如数学、计算机、通信、物理等。在第2章和第3章中，我们已经介绍了密码学中涉及到的信息理论和复杂度理论的基础知识。本节主要来介绍该书中所用到的一些数学知识。

1.1 概 率 论

一个试验是产生给定的结果集中的一个结果的过程。每一个可能的结果称为一个简单事件。所有可能的结果的集称为样本空间。根据密码学的需要，我们只考虑取有限多个可能的结果的离散样本空间。设样本空间为 S，S 中的简单事件为 $s_1, s_2, \cdots, s_n$。

定义 1.1.1 样本空间 S 上的一个概率分布是满足下列条件的一列实数 $p_1, p_2, \cdots, p_n$：

$$\sum_{i=1}^{n} p_i = 1$$

p_i 可解释为试验结果是 s_i 的概率。

定义 1.1.2 一个事件 E 是样本空间 S 的一个子集，事件 E 发生的概率记为 $p(E)$。特别地，当 E 是一个简单事件，即 $E=\{s_i\}$ 时，用 $p(s_i)$ 表示 $p(E)=p(\{s_i\})$。事件 E 发生的概率

$$p(E) = \sum_{s_i \in E} p(s_i)。$$

定义 1.1.3 如果 E 是一个事件，则 E 的补事件定义为 $\bar{E}=S\backslash E$。

关于事件 E 有下列基本性质：

(1) $0 \leqslant p(E) \leqslant 1$，$p(S)=1$，$p(\phi)=0$；

(2) $p(\bar{E})=1-p(E)$；

(3) 如果在 S 中的结果是等可能的，则 $p(E)=\dfrac{|E|}{|S|}$。

定义 1.1.4 设 E_1 和 E_2 是两个事件，如果 $p(E_1 \cap E_2)=0$，则称事件 E_1 和 E_2 互不相容。也就是说事件 E_1 和 E_2 不能同时发生。所谓一个事件 E 发生意指 E 中所包含的某一简单事件 s 在试验中出现。

关于两个事件 E_1 和 E_2 有下列基本性质：

(1) 如果 $E_1 \subseteq E_2$，则 $p(E_1) \leqslant p(E_2)$。

(2) $p(E_1 \cup E_2)+p(E_1 \cap E_2)=p(E_1)+p(E_2)$。特别地，当 E_1 和 E_2 互不相容时，$p(E_1 \cup E_2)=p(E_1)+p(E_2)$。

定义 1.1.5 设 E_1 和 E_2 是两个事件，$p(E_2)>0$，称 $p(E_1|E_2)=\dfrac{p(E_1\bigcap E_2)}{p(E_2)}$ 为在已知事件 E_2 发生的条件下，事件 E_1 发生的条件概率。

定义 1.1.6 如果 $p(E_1\bigcap E_2)=p(E_1)p(E_2)$，即 $p(E_1|E_2)=p(E_1)$ 或 $p(E_2|E_1)=p(E_2)$，则称事件 $E1$ 和 $E2$ 统计独立。

由定义 1.1.5 可知，$p(E_1\bigcap E_2)=p(E_2)p(E_1|E_2)$，这个公式称为概率的乘法公式，而

$$p(E_1 \bigcap E_2) = p(E_1)p(E_2|E_1)$$

由此可推出如下的 Bayes 公式：

$$p(E_1|E_2) = \frac{p(E_1)p(E_2|E_1)}{p(E_2)} \quad (p(E_2) > 0)$$

定义 1.1.7 设 S 是一个具有概率分布 p 的样本空间。一个随机变量 X 是一个从样本空间 S 到实数集 R 的函数；对每一个简单事件 $s_i\in S$，X 分配一个实数 $X(s_i)$。

因为我们这里假定 S 是有限的，所以 X 只能取有限个实数值。

定义 1.1.8 设 X 是 S 上的一个随机变量。X 的数学期望定义为

$$E(X) = \sum_{s_i\in S} X(s_i)p(s_i)$$

$E(X)$也可以改为

$$E(X) = \sum_{x\in R} xp(X = x)$$

随机变量的数学期望实质上反映了随机变量的平均值。

关于随机变量的数学期望有如下基本性质：

设 $X_1,X_2,\cdots,X_m$ 是 S 上的 m 个随机变量，$a_1,a_2,\cdots,a_m$ 是任意 m 个实数，则

$$E\Big(\sum_{i=1}^{m} a_iX_i\Big) = \sum_{i=1}^{m} a_iE(x_i)$$

定义 1.1.9 设 X 是 S 上的一个随机变量，X 的数学期望为 $\mu=E(X)$，则称 $D(X)=E((X-\mu)^2)$ 为随机变量 X 的方差。$D(X)$也可以表示为

$$D(X) = E(X^2) - (E(X))^2$$

方差 $D(X)$反映了 X 离开它的平均值 $E(X)$的偏离程度。

设随机变量 X 的期望值为 $\mu=E(X)$，方差为 $D(X)$，则对任意的 $\varepsilon>0$，事件($|X-\mu|\geqslant\varepsilon$)发生的概率 $p(|X-\mu|\geqslant\varepsilon)$与 $D(X)$的关系如下：

$$p(|X - \mu| \geqslant \varepsilon) \leqslant \frac{D(X)}{\varepsilon^2}$$

这就是著名的 Chebyshev 不等式。

1.2 数　论

用 Z 表示整数集$\{\cdots,-3,-2,-1,0,1,2,3,\cdots\}$。

定义 1.2.1 设 $a,b\in Z,a\neq0$，如果存在 $c\in Z$，使得 $b=ac$，则称 a 整除 b，并称 a 是 b 的因子，b 是 a 的倍数，记为 $a|b$。如果不存在整数 c，使得 $b=ac$，则称 a 不能整除 b，记为 $a\nmid b$。

例 1.2.1 $-3|18$，因为 $18=(-3)\times(-6)$。

由整除的定义不难推出数的整除性的一些基本性质：

(1)$a|a$。

(2)如果$a|b,b|c$，则$a|c$。

(3)如果$a|b,a|c$，则对所有的$x,y\in Z$，有$a|(bx+cy)$。

(4)如果$a|b,b|a$，则$a=\pm b$。

定理 1.2.1(带余除法) 设$a,b\in Z,b\geqslant 1$，则存在唯一确定的整数q和r，使得$a=qb+r,0\leqslant r<b$。q称为a除以b所得的商，r称为a除以b所得的余数，通常记$q=a \text{ div } b$，$r=a \bmod b$。

对任意的$a,b\in Z,b\neq 0$，显然$a \text{ div } b=[a/b]$，$a \bmod b=a-b[a/b]$。其中$[x]$表示不大于x的最大整数，例如$[5.2]=5,[-5.2]=-6$。

例 1.2.2 设$a=73,b=17$，则$q=4,r=5$，因此，$73 \bmod 17=5,73 \text{ div } 17=4$。

定义 1.2.2 设$a,b\in Z,a,b$不全为0，如果$c|a$且$c|b$，则称c为a和b的公因子。特别地，我们把a和b的所有公因子中最大的，称为a和b的最大公因子。

我们知道，每个非零整数只有有限多个因子，所以若a和b是两个不全为0的整数，则它们的公因子也只有有限多个，所以，它们的最大公因子必然存在，而且唯一。将这个最大公因子记为$\gcd(a,b)$。因为若c是a,b的公因子，则$-c$也一定是，所以$\gcd(a,b)>0$。当a,b全为0时，约定$\gcd(0,0)=0$。类似于两个整数的最大公因子的定义，我们可以定义任意有限多个整数的最大公因子。记为$\gcd(a_1,a_2,\cdots,a_n)=\gcd(\gcd(a_1,a,\cdots,a_{n-1}),a_n)$。

例 1.2.3 设$a=12,b=18$，则12和18的公因子为$\{-6,-3,-2,-1,1,2,3,6\}$，因此$\gcd(12,18)=6$。

定义 1.2.3 设$a,b\in Z,a,b$全不为0，如果$a|D,b|D$且$D\geqslant 1$，则称D为a和b的公倍数。特别地，我们把a和b的所有的公倍数中最小的正的公倍数，称为a和b的最小公倍数。

因为$|ab|$就是a和b的一个公倍数，所以a和b的最小公倍数一定存在而且唯一，将这个最小公倍数记为$\text{lcm}(a,b)$。两个非零整数的最小公倍数的概念可以推广到任意多个非零整数的情形。

易知，对两个正整数a和b，必有$ab=\gcd(a,b)\cdot\text{lcm}(a,b)$。

例 1.2.4 设$a=12,b=18$，则通过简单的计算知$\text{lcm}(a,b)=36$，显然

若不计因子的顺序，这个分解式是唯一的。其中 $p_i(1\leqslant i\leqslant k)$是不同的素数，$e_i(1\leqslant i\leqslant k)$是正整数。

设

$$a = p_1^{e_1}p_2^{e_2}\cdots p_k^{e_k} \qquad b = p_1^{f_1}p_2^{f_2}\cdots p_k^{f_k} \qquad (e_i \geqslant 0, f_i \geqslant 0, 1 \leqslant i \leqslant k)$$

则

$$\gcd(a,b) = p_1^{\min(e_1,f_1)}p_2^{\min(e_2,f_2)}\cdots p_k^{\min(e_k,f_k)}$$
$$\mathrm{lcm}(a,b) = p_1^{\max(e_1,f_1)}p_2^{\max(e_2,f_2)}\cdots p_k^{\max(e_k,f_k)}$$

例 1.2.5 设

$$a = 4864 = 2^8 \times 19 \qquad b = 3458 = 2 \times 7 \times 13 \times 19$$

则

$$\gcd(4864,3458) = 2 \times 19 \qquad \mathrm{lcm}(4864,3458) = 2^8 \times 7 \times 13 \times 19$$

定义 1.2.5 设 $n\geqslant 1$，$\varphi(n)$表示在区间$[1,n]$中与 n 互素的整数的数目，函数 $\varphi(n)$称为 Euler 函数。

关于 Euler 函数有以下性质：

(1) 如果 p 是素数，则 $\varphi(p)=p-1$。

(2) Euler 函数是一个积性函数，也就是说，如果 $\gcd(m,n)=1$，则 $\varphi(mn)=\varphi(m)\varphi(n)$。

(3) 如果 $n=p_1^{e_1}p_2^{e_2}\cdots p_k^{e_k}$ 是 n 的一个典型分解式，则

$$\varphi(n) = n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_k}\right)$$

(4) 对于所有的 $n\geqslant 5$，有$\varphi(n)\geqslant n/6\ln\ln n$。

虽然我们可以通过分解两个正整数 a 和 b 来计算它们的最大公因子，但是目前还没有分解整数的有效算法。这里我们来描述一个计算两个整数的最大公因子的有效算法，称为 Euclidean 算法，其理论依据是：如果 a,b 是两个正整数，$a>b$，则

$$\gcd(a,b) = \gcd(b,a \bmod b)$$

计算两个整数的最大公因子的 Euclidean 算法

输入：两个非负整数 a 和 b，$a\geqslant b$。

输出：a 和 b 的最大公因子。

(1)当 $b\neq 0$ 时，完成下列步骤：

置 $r\leftarrow a \bmod b$，$a\leftarrow b$，$b\leftarrow r$。

(2)返回 a。

该算法的运行时间为 $O((\lg n)^2)$。

例 1.2.6 设 $a=4864$，$b=3458$，按上述算法计算 gcd(4864,3458)的步骤如下：

$$4864=1\times 3458+1406$$
$$3458=2\times 1406+646$$
$$1406=2\times 646+114$$
$$646=5\times 114+76$$
$$114=1\times 76+38$$
$$76=2\times 38+0$$

故 gcd(4864,3458)=38。

Euclidean 算法能被扩展使得不仅可用来计算整数 a 和 b 的最大公因子 d，而且可以找到满足下式的整数 x 和 y：$ax+by=d$。

扩展的 Euclidean 算法

输入：两个非负整数 a 和 b，$a\geqslant b$。

输出：$d=\gcd(a,b)$和满足等式 $ax+by=d$ 的整数 x 和 y。

(1)如果 $b=0$，则置 $d\leftarrow a,x\leftarrow 1,y\leftarrow 0$，并返回$(d,x,y)$。

(2)置 $x_2\leftarrow 1,x_1\leftarrow 0,y_2\leftarrow 0,y_1\leftarrow 1$。

(3)当 $b>0$ 时，执行下列步骤：

$q\leftarrow[a/b],r\leftarrow a-qb,x\leftarrow x_2-qx_1,y\leftarrow y_2-qy_1$。

$a\leftarrow b,b\leftarrow r,x_2\leftarrow x_1,y_2\leftarrow y_1,x_1\leftarrow x,y_1\leftarrow y$。

(4)置 $d\leftarrow a,x\leftarrow x_2,y\leftarrow y_2$，并返回$(d,x,y)$。

该算法的时间复杂度为 $O((\lg n)^2)$。

例 1.2.7 设 $a=4864,b=3458$。下表是用扩展 Euclidean 算法计算 $\gcd(a,b)$的各步骤：

q	r	x	y	a	b	x_2	x_1	y_2	y_1
				4864	3458	1	0	0	1
1	1406	1	−1	3458	1406	0	1	1	−1
2	646	−2	3	1406	646	1	−2	−1	3
2	114	5	−7	646	114	−2	5	3	−7
5	76	−27	38	114	76	5	−27	−7	38
1	38	32	−45	76	38	−27	32	38	−45
2	0	−91	128	38	0	32	−91	−45	128

上表表明

$$\gcd(4864,3458)=38,4864\times 32+3458\times(-45)=38$$

同余式

设 n 是一个正整数。

定义 1.2.6 设 $a,b\in Z$，如果 $n|(a-b)$，则我们说 a 和 b 模 n 同余，记为 $a\equiv b\pmod n$。整数 n 称为同余模。

例 1.2.8 (1) $24\equiv 9\pmod 5$，因为 $24-9=3\times 5$。

(2) $-11\equiv 17\pmod 7$，因为 $-11-17=(-4)\times 7$。

同余式具有下列一些基本性质：

(1) $a\equiv b\pmod n$，当且仅当 $a \bmod n = b \bmod n$。

(2)(反身性) $a\equiv a\pmod n$。

(3)(对称性) 如果 $a\equiv b\pmod n$，那么 $b\equiv a\pmod n$。

(4)(传递性)如果 $a\equiv b\pmod n$，$b\equiv c\pmod n$，那么 $a\equiv c\pmod n$。

(5) 如果 $a\equiv a_1\pmod n$，$b\equiv b_1\pmod n$，那么，$a+b\equiv a_1+b_1\pmod n$，$ab\equiv a_1b_1\pmod n$。

一个整数 a 的等价类是所有与 a 模 n 同余的整数所作成的集合。由性质(2)～(4)可

知，对一个固定的 n，模 n 同余关系将 Z 分成等价类。如果 $a=qn+r, 0\leqslant r<n$，那么 $a\equiv r \pmod n$。因此每一个整数 a 与 0 至 $n-1$ 之间唯一的一个整数模 n 同余。0 至 $n-1$ 之间的这一整数称为 a 模 n 的最小剩余。这样 a 和 r 在同一个等价类之中，并且用 r 来表示这个等价类，也称剩余类。

用 Z_n 来表示模 n 的整数之集，它是整数集 $\{0,1,2,\cdots,n-1\}$，也就是模 n 的等价类之集。Z_n 中的加法和乘法均为模 n 运算。

例 1.2.9 设 $Z_{25}=\{0,1,2,\cdots 24\}$。在 Z_{25} 中，$13+16=4$，因为 $13+16=29\equiv 4 \pmod{25}$。类似地在 Z_{25} 中，$13\times 16=8$。

定义 1.2.7 设 $a\in Z_n$，如果存在 $x\in Z_n$ 使得 $ax\equiv 1 \pmod n$ 那么我们就说 a 是可逆的，称 x 为 a 的逆元。如果 a 是可逆的，那么容易证明其逆元是唯一的，记为 a^{-1}。

设 $a,b\in Z_n$，b 是可逆的，则可在 Z_n 中定义除法：$a/b=ab^{-1} \pmod n$。注意只有存在逆元的 b，才可定义 a/b。易证 $a\in Z_n$ 是可逆的，当且仅当 $\gcd(a,n)=1$。由前面的讨论可知，Z_n 中共有 $\varphi(n)$ 个可逆元。

例 1.2.10 Z_9 中的可逆元是 1,2,4,5,7,8。$4^{-1}=7$，因为 $4\times 7\equiv 1 \pmod 9$。

令 $Z_n^*=\{a\in Z_n \mid \gcd(a,n)=1\}$，则 $|Z_n^*|=\varphi(n)$。特别地，当 n 是一个素数时，$Z_n^*=\{1,2,\cdots,n-1\}$。可证明 Z_n^* 在模 n 的乘法运算下形成一个群（群的定义本见附录 1.3 节）。

关于 Z_n^* $(n\geqslant 2)$ 有如下的事实：

(1) (Euler 定理) 如果 $a\in Z_n^*$，那么 $a^{\varphi(n)}\equiv 1 \pmod n$。特别地，当 n 是一个素数时，$a^{n-1}\equiv 1 \pmod n$，这就是著名的 Fermat 定理。

(2) 如果 $a\in Z_n^*$，那么可用扩展的 Euclidean 算法找到整数 x 和 y 使得 $ax+by=1$，这样 $a^{-1}=x \pmod n$。

中国剩余定理（孙子定理）：设 $m_i\in Z, 1\leqslant i\leqslant r, \gcd(m_i,m_j)=1, i\neq j$。$a_i\in Z, 1\leqslant i\leqslant r$。考虑下列的同余方程组：

$$\begin{cases} x\equiv a_1 \pmod{m_1} \\ x\equiv a_2 \pmod{m_2} \\ \vdots \\ x\equiv a_r \pmod{m_r} \end{cases}$$

中国剩余定理（又称孙子定理）说明了上述方程组模 $M=m_1m_2\cdots m_r$ 有唯一的解。

定理 1.2.2（中国剩余定理） 假定 $m_1,m_2,\cdots,m_r$ 是两两互素的 r 个整数，$a_1,a_2,\cdots,a_r$ 是任意 r 个整数，则 r 个同余方程 $x\equiv a_i \pmod{m_i}$ $(1\leqslant i\leqslant r)$ 组成的同余方程组模 $M=m_1m_2\cdots m_r$ 有唯一的一个解。该解可由下式给出：

$$x=\sum_{i=1}^{r} a_iM_iy_i \mod M$$

其中 $M_i=M/m_i$，$y_i=M_i^{-1} \bmod m_i$，$1\leqslant i\leqslant r$。

二次剩余

定义 1.2.8 设 $a\in Z_n^*$，如果存在一个 $x\in Z_n^*$ 使得 $x^2\equiv a \pmod n$，那么我们称 a 是一个模 n 的二次剩余，否则，我们称 a 是一个模 n 的二次非剩余。记所有模 n 的二次剩余之集为 Q_n，记所有模 n 的二次非剩余之集为 $\bar{Q}_n$。

由定义可知，$0\notin Z_n^*$，因此，$0\notin Q_n$，$0\notin \bar{Q}_n$。

关于二次剩余有以下一些基本事实：

(1) 当 $n=p$ 是一个奇素数时，$|Q_p|=|\bar{Q}_p|=(p-1)/2$。

(2) 当 $n=pq$，p 和 q 是两个不同的奇素数时：

$$|Q_n|=|Q_p||Q_q|=(p-1)(q-1)/4,|\bar{Q}_n|=3(p-1)(q-1)/4$$

(3) (Euler 准则) 设 p 是一个奇素数，则 $x\in Q_p$，当且仅当 $x^{(p-1)/2}\equiv 1(\mathrm{mod}\ p)$。

设 $a\in Q_n$，如果 $x\in Z_n^*$ 使得 $x^2\equiv a(\mathrm{mod}\ n)$，那么也称 x 是 a 模 n 的一个平方根。

Legendre 和 Jacobi 符号

定义 1.2.9 设 p 是一个奇素数，a 是一个整数。则 Legendre 符号 $\left(\frac{a}{p}\right)$ 定义为

$$\left(\frac{a}{p}\right)=\begin{cases}0 & p|a\\ 1 & a\in Q_p\\ -1 & a\in\bar{Q}_p\end{cases}$$

Legendre 符号有以下的性质：

(1) $\left(\frac{a}{p}\right)\equiv a^{(p-1)/2}(\mathrm{mod}\ p)$。特别地，$\left(\frac{1}{p}\right)=1$，$\left(\frac{-1}{p}\right)=(-1)^{(p-1)/2}$。因此，如果 $p\equiv 1\ (\mathrm{mod}\ 4)$，那么 $-1\in Q_p$；如果 $p\equiv 3\ (\mathrm{mod}\ 4)$，那么 $-1\in\bar{Q}_p$。

(2) $\left(\frac{ab}{p}\right)=\left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$。因此，如果 $a\in Z_p^*$，那么 $\left(\frac{a^2}{p}\right)=1$。

(3) 如果 $a\equiv b(\mathrm{mod}\ p)$，那么 $\left(\frac{a}{p}\right)=\left(\frac{b}{p}\right)$。

(4) $\left(\frac{2}{p}\right)=(-1)^{(p^2-1)/8}$。因此，如果 $p\equiv 1$ 或 $7(\mathrm{mod}\ 8)$，那么 $\left(\frac{2}{p}\right)=1$；如果 $p\equiv 3$ 或 $5(\mathrm{mod}\ 8)$，那么 $\left(\frac{2}{p}\right)=-1$。

(5) (二次互反律)设 q 是一个不同于 p 的奇素数，则 $\left(\frac{p}{q}\right)=\left(\frac{q}{p}\right)(-1)^{(p-1)(q-1)/4}$。

Jacobi 符号是 Legendre 符号的一个一般化，它不只局限于 n 是奇素数，n 是奇数也可以。

定义 1.2.10 设 $n\geqslant 3$ 是一个奇数，$n=p_1^{e_1}p_2^{e_2}\cdots p_k^{e_k}$，则 Jacobi 符号 $\left(\frac{a}{n}\right)$ 定义为：

$$\left(\frac{a}{n}\right)=\left(\frac{a}{p_1}\right)^{e_1}\left(\frac{a}{p_2}\right)^{e_2}\cdots\left(\frac{a}{p_k}\right)^{e_k}$$

特别地，当 n 是奇素数时，Jacobi 符号就是 Legendre 符号。

Jacobi 符号有以下基本性质：

(1) $\left(\frac{a}{n}\right)=0,1$ 或 -1，再者，$\left(\frac{a}{n}\right)=0$，当且仅当 $\gcd(a,n)\neq 1$。

(2) $\left(\frac{ab}{n}\right)=\left(\frac{a}{n}\right)\left(\frac{b}{n}\right)$，因此，如果 $a\in Z_n^*$，那么 $\left(\frac{a^2}{n}\right)=1$。

(3) $\left(\frac{a}{mn}\right)=\left(\frac{a}{m}\right)\left(\frac{a}{n}\right)$。

(4) $\left(\frac{1}{n}\right)=1$。

(5) 如果 $a\equiv b(\mathrm{mod}\ n)$，那么 $\left(\frac{a}{n}\right)=\left(\frac{b}{n}\right)$。

(6) $\left(\frac{-1}{n}\right)=(-1)^{(n-1)/2}$。

(7) $\left(\frac{2}{n}\right)=(-1)^{(n^2-1)/8}$。

(8)(二次互反律)$\left(\frac{m}{n}\right)=\left(\frac{n}{m}\right)(-1)^{(m-1)(n-1)/4}$。

例 1.2.11 计算$\left(\frac{7411}{9283}\right)$。

$$\left(\frac{7411}{9283}\right)=(-1)^{\frac{(7411-1)(9283-1)}{4}}\left(\frac{9283}{7411}\right)=-\left(\frac{1872}{7411}\right)=-\left(\frac{2}{7411}\right)^2\left(\frac{3}{7411}\right)^2\left(\frac{13}{7411}\right)$$

$$=-\left(\frac{13}{7411}\right)=(-1)(-1)^{\frac{(13-1)(7411-1)}{4}}\left(\frac{7411}{13}\right)=-\left(\frac{1}{13}\right)=-1。$$

1.3 代数基础

本节我们简要介绍三种最基本的代数结构,群,环,域。

1.3.1 群

定义 1.3.1 设 S 是一个集合,在 S 上的一个二元运算 $*$ 是一个从 $S\times S$ 到 S 的映射。

定义 1.3.2 一个群$(G,*)$由一个满足下列三个条件的二元运算 $*$ 构成:

(1) 群运算是可结合的。也就是说,对所有的 a,b,c,有 $a*(b*c)=(a*b)*c$。

(2) 有一个称作单位的元素 $1\in G$,使得对所有的 $a\in G$,有 $a*1=1*a=a$。

(3) 对每一个 $a\in G$,有一个称作 a 的逆的元素 b 使得 $b*a=a*b=1$,把 b 记为 a^{-1}。

一个群 G 是 Abelian 群或交换群或加法群,如果它还满足下列条件:

(4) 对所有的 $a,b\in G$ 有 $a*b=b*a$。

对加法群而言,通常用 0 来表示单位元素,用 $-a$ 表示 a 的逆。

定义 1.3.3 如果 $|G|$ 是有限的,那么称群 G 是有限群。

一个有限群的元素的个数称为它的阶。

例 1.3.1 整数集 Z 关于普通的加法形成一个 Abelian 群,单位是 0,a 的逆是 $-a$。

例 1.3.2 剩余类集 Z_n 关于模 n 加法运算形成一个阶为 n 的 Abelian 群,Z_n 关于模 n 的乘法运算不是一个群,因为不是所有的元素都有乘法逆。然而,集 Z_n^* 关于模 n 的乘法运算形成一个阶为$\varphi(n)$的 Abelian 群,单位元素为 1。

定义 1.3.4 设 H 是群$(G,*)$的一个非空子集,如果 H 本身在群 G 的运算 $*$ 之下构成一个群,我们说 H 是 G 的一个子群。如果 H 是 G 的一个子群且 $H\neq G$,则称 H 是 G 的一个真子群。

定义 1.3.5 设 G 是一个群,如果 G 中有一个元素 a 使得对每一个 $b\in G$ 都存在一个整数 i 使得 $b=a^i$,则称 G 是一个循环群,a 称为 G 的一个生成元。

定义 1.3.6 设 G 是一个群,$a\in G$,a 的阶定义为使得 $a^t=1$ 的最小正整数 t(假定这样的正整数存在的话)。如果这样的正整数 t 不存在,那么 a 的阶定义为∞。

易知,如果 G 是一个群,$a\in G$,则$\{a^k\mid k\in Z\}$形成 G 的一个循环子群,称作由 a 生成的子群,记为$\langle a\rangle$。如果 a 的阶为 t,则$|\langle a\rangle|=t$。

关于有限群有下列重要的基本定理:

定理 1.3.1(Lagrange 定理) 设 G 是一个有限群，H 是 G 的一个子群，则 $|H| \mid |G|$。因此，如果 $a \in G$，则 a 的阶整除 $|G|$。

关于循环群有以下基本性质：

(1)循环群的子群也是循环群。如果 G 是一个阶为 n 的循环群，那么对 n 的每一个正因子 d，G 恰好包含一个阶为 d 的子群。

(2)设 G 是一个群，如果 $a \in G$ 的阶是 t，则 a^k 的阶是 $t/\gcd(t,k)$。

(3)设 G 是一个阶为 n 的循环群，$d \mid n$，则 G 恰有 $\varphi(d)$ 个阶为 d 的元素。特别地，G 有 $\varphi(n)$ 个生成元。

1.3.2 环

定义 1.3.7 一个环 $(R,+,\times)$ 是由两个满足下列条件的 R 上的二元运算“+”(称为加法)和“×”(称为乘法)构成：

(1) $(R,+)$ 是一个 Abelian 群，单位用 0 表示。

(2) 运算“×”是可结合的，也就是说，对所有的 $a,b,c \in R$，有 $a\times(b\times c)=(a\times b)\times c$。

(3) 有一个乘法单位 $1,1\neq 0$，使得对所有的 $a \in R$，有 $a\times 1=1\times a=a$。

(4) 乘法和加法由分配律联系着，也就是对所有的 $a,b,c \in R$，有 $a\times(b+c)=(a\times b)+(a\times c)$ 和 $(b+c)\times a=(b\times a)+(c\times a)$。

如果一个环 $(R,+,\times)$ 还满足条件：对所有的 $a,b \in R$，有 $a\times b=b\times a$，则称环 $(R,+,\times)$ 为交换环。

例 1.3.3 整数集 Z 关于通常的加法和乘法运算形成一个交换环。

例 1.3.4 Z_n 关于模 n 加法和乘法形成一个交换环。

定义 1.3.8 设 R 是一个环，$a \in R$，如果存在一个元素 $b \in R$ 使得 $a\times b=1$，则称 a 是一个单位或可逆元素。

环 R 的所有单位之集关于乘法形成一个群，称为 R 的单位群。

例 1.3.5 环 Z_n 的单位群是 Z_n^*。

1.3.3 域

定义 1.3.9 满足下列条件的一个交换环称为一个域：所有的非零元素都有乘法逆。

定义 1.3.10 设 F 是一个域，如果对任何 $m \geqslant 1$，$1+1+\cdots+1\neq 0$ (m 个 1 相加)，则称 F 的特征为 0，否则，F 的特征是使得 $\sum_{i=1}^{m} 1=0$ 的最小正整数 m。

例 1.3.6 整数环 Z 不是一个域，因为只有 1 和 -1 有乘法逆。然而，有理数集 Q、实数集 R 和复数集 C 在通常的加法和乘法运算下都形成特征为 0 的域。

定理 1.3.2 Z_n 是一个域，当且仅当 n 是素数。如果 n 是素数，则 Z_n 的特征是 n。

关于域的特征有以下重要事实：如果一个域的特征不是 0，则它的特征必是素数。

定义 1.3.11 设 E 是一个域，F 是 E 的一个子集，如果 F 关于 E 的运算本身形成一个域，则称 F 为 E 的子域，也称 E 为 F 的扩域。

根据域所包含的元素是否有限，将域分为无限域和有限域；包含有限个元素的域称为有限域，否则称为无限域。例如有理数域 Q 是一个无限域，Z_p (p 为素数)是一个有限域，域

F 中的元素的个数也称为有限域 F 的阶。

关于有限域有以下的重要结果：

(1)(有限域的存在性和唯一性) 如果 F 是一个有限域，那么 F 包含素数幂个元素，即存在素数 p 和整数 $m\geqslant 1$，使得 $|F|=p^m$；反之，对每一个素数幂 p^m，(在同构的意义下)存在唯一的一个阶为 p^m 的有限域。将这个域记为 F_{p^m} 或 $GF(p^m)$。

(2)(有限域的子域)设 F_q 是一个阶为 $q=p^m$ 的有限域，p 为素数，则 F_q 的每个子域有阶 p^n，且 $n|m$。反之，如果 $n|m$，则恰有 F_q 的一个阶为 p^n 的子域，而且 $a\in F_q$ 是 F_{p^n} 的一个元素，当且仅当 $a^{p^n}=a$。

(3)$F_q^*=F_q\backslash\{0\}$ 关于乘法形成一个阶为 $q-1$ 的循环群。因此，对所有的 $a\in F_q$，有 $a^q=a$。这个群称为 F_q 的乘法群，乘法群 F_q^* 的生成元称为 F_q 的本原元，共有 $\varphi(q-1)$ 个本原元。

(4)设 F_q(其中 $q=p^m$)是一个有限域，p 是一个素数，$m\geqslant 1$，则 F_q 的特征为 p，而且对所有的 $a,b\in F_q$ 和 $t\geqslant 0$，有 $(a+b)^{p^t}=a^{p^t}+b^{p^t}$。

1.3.4 向量空间

定义 1.3.12 在一个域 F 上的向量空间 V 是由一个 Abelian 群 $(V,+)$ 和一个满足下列条件的乘法运算“$\cdot$”：$F\times V\to V$ 构成：

$$(1)\ a(v+w)=av+aw$$
$$(2)\ (a+b)v=av+bv$$
$$(3)\ (ab)v=a(bv)$$
$$(4)\ 1v=v$$

其中 $a,b\in F$，$v,w\in V$。

V 中的元素称为向量，F 中的元素称为标量。群运算“+”称为向量的加法，乘法“$\cdot$”称为标量乘法。

定义 1.3.13 设 V 是域 F 的一个向量空间，V 的一个子空间 U 是满足下列条件的 V 的一个加法子群：对所有的 $a\in F$ 和 $u\in U$，有 $au\in U$。

显然，向量空间的一个子空间也是一个向量空间。

定义 1.3.14 设 $S=\{v_1,v_2,\cdots,v_n\}$ 是域 F 上的向量空间 V 的一个子集。

(1)S 的一个线性组合是一个形式为 $a_1v_1+a_2v_2+\cdots a_nv_n$ 的表达式，$a_i\in F$，$1\leqslant i\leqslant n$。

(2)令 $\langle S\rangle=\{a_1v_1+a_2v_2+\cdots+a_nv_n\,|\,a_i\in F,1\leqslant i\leqslant n\,|\}$，称 $\langle S\rangle$ 为由 S 张成的向量空间，它是 V 的一个子空间。

(3) 如果存在不全为 0 的标量 $a_1,a_2,\cdots,a_n$ 使得 $a_1v_1+a_2v_2+\cdots a_nv_n=0$，我们称 S 在 F 上是线性相关的，否则，称 S 在 F 上是线性独立的或线性无关的。

(4) 如果 S 是线性独立的，而且 $V=\langle S\rangle$，则称 S 为 V 的一组基。$|S|$ 称为 V 的维数。

例 1.3.7 设 F 是任何一个域，$V=F^n=F\times F\times\cdots\times F$ 是 F 的 n-重笛卡尔积，则 V 是 F 上的一个 n 维向量空间。$S=\{e_1,e_2,\cdots,e_n\}$ 为 V 的一个基，其中 e_i 的第 i 个分量为 1，其余分量为 0。

1.3.5 多项式环

设 R 是一个交换环，x 是一个未定元。设 i 是一个非负整数，形如 $a_ix^i(a_i\in R)$ 的式子，叫做系数属于 R 的 x 的单项式。而有限个系数属于 R 的单项式 $a_0x^0,a_1x^1,\cdots,a_nx^n$（其中 n 是任意非负整数，$a_i\in R,0\leqslant i\leqslant n$）的形式和

$$a_0x^0+a_1x^1+\cdots+a_nx^n \tag{1.3.1}$$

就叫做系数属于 R 的 x 的多项式，或简称环 R 上的 x 的多项式。在多项式(1.3.1)中，a_ix^i 叫做它的 i 次项，a_i 叫做它的 i 次项的系数。约定 $a_0x^0=a_0$，并将 x^1 记为 x，那么多项式(1.3.1)又可写成 $a_0+a_1x+\cdots+a_nx^n$。我们通常用记号 $f(x),g(x),h(x),f_i(x),\cdots$ 等来表示多项式。

设 $f(x)$ 和 $g(x)$ 是 R 上的 x 的两个多项式。如果除去系数等于 R 中的零元素的项以外，它们同次项的系数都相等，我们就说 $f(x)$ 和 $g(x)$ 相等，记作 $f(x)=g(x)$。通常可将多项式 $f(x)=a_0+a_1x+\cdots+a_nx^n$ 表示为 $f(x)=\sum\limits_{i=0}^{n}a_ix^i$。如果 $a_n\neq0$，我们就说 $f(x)$ 是 n 次多项式，记作 $\partial^0(f(x))=n$，并称 a_n 为 $f(x)$ 的首项系数。当 $f(x)$ 的所有系数都是 0 时，我们就说 $f(x)$ 是零多项式，仍用 0 来代表它，约定 $\partial^0(0)=-\infty$。

记 R 上的 x 的多项式的全体所成之集为 $R[x]$。现在我们在 $R[x]$ 中定义加法和乘法运算。设

$$f(x)=\sum_{i=0}^{n}a_ix^i \qquad g(x)=\sum_{i=0}^{m}b_ix^i\in R[x]$$

令 $M=\max(n,m)$，当 $n<m$ 时，约定 $a_{n+1}=a_{n+2}=\cdots=a_m=0$。当 $m<n$ 时，约定 $b_{m+1}=b_{m+2}=\cdots=b_n=0$。此时，$f(x)$ 和 $g(x)$ 可分别写成

$$f(x)=\sum_{i=0}^{M}a_ix^i \qquad g(x)=\sum_{i=0}^{M}b_ix^i$$

定义加法运算：$f(x)+g(x)=\sum\limits_{i=0}^{M}(a_i+b_i)x^i$，显然 $f(x)+g(x)\in R[x]$。再令 $a_{n+1}=a_{n+2}=\cdots=a_{m+n}=0,b_{m+1}=b_{m+2}=\cdots=b_{n+m}=0$，此时，$f(x)$ 和 $g(x)$ 可分别写成

$$f(x)=\sum_{i=0}^{m+n}a_ix^i,g(x)=\sum_{i=0}^{m+n}b_ix^i$$

定义乘法运算：$f(x)\cdot g(x)=\sum\limits_{i=0}^{m+n}\left(\sum\limits_{j=0}^{i}a_jb_{i-j}\right)x^i$，显然 $f(x)\cdot g(x)\in R[x]$。

易知

$$\partial^0(f(x)+g(x))\leqslant\max(\partial^0(f(x)),\partial^0(g(x)))$$

$$\partial^0(f(x)\cdot g(x))=\partial^0(f(x))+\partial^0(g(x))$$

直接可验证 $(R[x],+,\cdot)$ 是一个交换环，通常称为多项式环。

例 1.3.8 设 $R=Z_2$

$$f(x)=x^3+x+1 \qquad g(x)=x^2+x\in Z_2[x]$$

$$f(x)+g(x)=x^3+x^2+1,f(x)\cdot g(x)=x^5+x^4+x^3+x$$

下面我们取 $R=F$，F 是一个任意的域。多项式环 $F[x]$ 与整数环 Z 非常类似，有许多共同的特性。下面我们来讨论这些类似的特性。

对应于整数环 Z 中的素数的概念，多项式环 $F[x]$ 中引入了不可约多项式的概念，它的作用类似于素数的作用。

定义 1.3.15 设 $f(x)\in F[x]$，$\partial^0(f(x))\geqslant 1$。如果 $f(x)$ 不能写成 $F[x]$ 中的两个次数不小于 1 的多项式之积，则我们称 $f(x)$ 在 F 上不可约。

定理 1.3.3(带余除法) 设 $a(x),b(x)\in F[x]$，$b(x)\neq 0$，则在 $F[x]$ 中存在唯一的一对多项式 $q(x)$ 和 $r(x)$ 使得 $a(x)=q(x)b(x)+r(x)$，$\partial^0(r(x))<\partial^0(b(x))$。多项式 $q(x)$ 称作商式，$r(x)$ 称作余式，通常记 $r(x)=a(x)\bmod b(x)$，$q(x)=a(x)\operatorname{div} b(x)$。当 $r(x)=0$ 时，我们就说 $b(x)$ 整除 $a(x)$，并称 $b(x)$ 是 $a(x)$ 的因式，或 $a(x)$ 是 $b(x)$ 的倍式，记为 $b(x)\mid a(x)$。当 $r(x)\neq 0$ 时，我们就说 $b(x)$ 不整除 $a(x)$，记为 $b(x)\nmid a(x)$。

例 1.3.9 设

$$a(x)=x^6+x^5+x^3+x^2+x+1 \quad b(x)=x^4+x^3+1\in Z_2[x]$$

则

$a(x)=x^2b(x)+x^3+x+1$，因此，$a(x)\operatorname{div} b(x)=x^2$，$a(x)\bmod b(x)=x^3+x+1$。

类似于整数环 Z 的情形，可在 $F[x]$ 中定义公因式，最高公因式和最低公倍式等概念。这里我们只讨论最高公因式这一概念。设 $a(x),b(x),c(x)\in F[x]$，$c(x)\neq 0$，$a(x)$ 和 $b(x)$ 不全为 0。如果 $c(x)$ 既是 $a(x)$ 的因式，又是 $b(x)$ 的因式，我们就说 $c(x)$ 是 $a(x)$ 和 $b(x)$ 的公因式。我们把 $a(x)$ 和 $b(x)$ 的公因式中的次数最高的而且首项系数等于 1 的公因式 $d(x)$ 叫做 $a(x)$ 和 $b(x)$ 的最高公因式，记为 $\gcd(a(x),b(x))$。约定 $\gcd(0,0)=0$。

当 $\gcd(a(x),b(x))=1$ 时，我们就说 $a(x)$ 和 $b(x)$ 互素。象整数环 Z 中一样，求两个不全为 0 的多项式的最高公因式也有相应的 Euclidean 算法和扩展的 Euclidean 算法。平行于算术基本定理，在 $F[x]$ 中也有相应的定理，称为唯一因式分解定理，这里不再赘述。

最后我们看一看多项式的剩余类或同余类环。设 $f(x)\in F[x]$ 是一个固定的多项式。

定义 1.3.16 设 $g(x),h(x)\in F[x]$，如果 $f(x)\mid(g(x)-h(x))$，则称 $g(x)$ 和 $h(x)$ 关于模 $f(x)$ 同余，记为 $g(x)\equiv h(x)\bmod(f(x))$。

$F[x]$ 中的同余式和 Z 中的同余式具有相同的性质。对一个固定的 $f(x)\in F[x]$。$g(x)\in F[x]$ 的等价类是所有与 $g(x)$ 模 $f(x)$ 同余的多项式所作成的集合。模 $f(x)$ 同余关系将 $F[x]$ 分成等价类。如果 $g(x)\in F[x]$，那么

$$g(x)=q(x)f(x)+r(x),\partial^0(r(x))<\partial^0(f(x)),q(x),r(x)\in F[x]$$

显然，$g(x)\equiv r(x)\bmod(f(x))$。用 $r(x)$ 来表示包含 $g(x)$ 的等价类。

用 $F[x]/(f(x))$ 表示 $F[x]$ 中次数小于 $n=\partial^0(f(x))$ 的全体多项式之集，也就是模 $f(x)$ 的等价类之集。$F[x]/(f(x))$ 在模 $f(x)$ 的加法和乘法下形成一个交换环，称为多项式剩余类环。

定理 1.3.4 如果 $f(x)\in F[x]$ 在 F 上不可约，则 $F[x]/(f(x))$ 是一个域。

由定理 1.3.4 可知，如果我们取 $F=Z_p$（p 为一个素数），$f(x)$ 是一个 m 次不可约多项式，则 $F[x]/(f(x))$ 是一个阶为 p^m 的有限域。这表明，可用不可约多项式来构造有限域。那么不可约多项式是否存在呢？人们已经证明，对任意的有限域 F 和任意的正整数 n，$F[x]$ 中一定存在 n 次不可约多项式，不仅如此，人们还给出了不可约多项式的精确计数公式。

2. AES 候选算法简介

1997 年 4 月 15 日美国国家标准技术研究所(NIST)发起征集 AES(AES—Advanced Encryption Standard)算法的活动,并专门成立了 AES 工作组,目的是为了确定一个非保密的、公开披露的、全球免费使用的分组密码算法,用于保护下一世纪政府的敏感信息,也希望能够成为秘密和公开部门的数据加密标准(DES)。1997 年 9 月 12 日在联邦登记处(FR)公布了征集 AES 候选算法的通告。AES 的基本要求是比三重 DES 快而且至少和三重 DES 一样安全,分组长度为 128 比特,密钥长度为 128/192/256 比特。1998 年 8 月 20 日 NIST 召开了第一次 AES 候选会议,并公布了 15 个 AES 候选算法,目前正处在评论阶段。除了 NIST 对这些候选算法做大量的分析和测试之外,也欢迎每个对此感兴趣的部门和个人提出自己的看法和意见。1999 年 3 月 22 日将举行第二次 AES 候选会议,公开 15 个候选算法的讨论结果。关于这些算法的讨论将于 1999 年 4 月 15 日结束,到时将从这 15 个算法中选出 5 个。NIST 声称最终将在这 5 个算法中遴选出一个算法作为 AES,预计于 2001 年出台 AES。本节简要介绍这些算法的基本设计思想和观点。

2.1 CAST-256

CAST-256 算法是在已有的 CAST-128 算法的基础上设计的一个算法,它的分组长度为 128 比特,密钥长度为 256 比特(当然密钥长度也可以取 128、160、192、224 比特),轮数为 48 即 12 个 4 轮。该算法使用了以下几种运算:(1)模 2^{32} 加法和减法运算,分别用"+"和"−"表示。(2)逐比特异或运算,用"⊕"表示。(3)循环左移运算,用"<<<"表示。

加密过程:输入分组放入一个长度为 128 比特的寄存器 x 中,输出分组仍放在寄存器 x 之中。设 x="128 比特的明文"$=ABCD$,A、B、C、D 均为 32 比特长,则

$$\text{for } (i=0;i<6;i++)$$
$$x \leftarrow Q_i(x)$$
$$\text{for } (i=6;i<12;i++)$$
$$x \leftarrow \bar{Q}_i(x)$$

最后在 x 中存储的 128 比特消息即为密文。

其中"$x \leftarrow Q_i(x)$"是下列赋值过程的简写:

$$C = C \oplus f_1(D, k_{r_0}^{(i)}, k_{m_0}^{(i)}) \qquad B = B \oplus f_2(C, k_{r_1}^{(i)}, k_{m_1}^{(i)})$$
$$A = A \oplus f_3(B, k_{r_2}^{(i)}, k_{m_2}^{(i)}) \qquad D = D \oplus f_1(A, k_{r_3}^{(i)}, k_{m_3}^{(i)})$$

"$x \leftarrow \bar{Q}_i(x)$"是下列赋值过程的简写:

$$D = D \oplus f_1(A, k_{r_3}^{(i)}, k_{m_3}^{(i)}) \qquad A = A \oplus f_3(B, k_{r_2}^{(i)}, k_{m_2}^{(i)})$$
$$B = B \oplus f_2(C, k_{r_1}^{(i)}, k_{m_1}^{(i)}) \qquad C = C \oplus f_1(D, k_{r_0}^{(i)}, k_{m_0}^{(i)})$$

下面我们来描述加密过程中涉及到的三个函数 f_1,f_2,f_3 的工作过程。

$f_1(y, k_{r_i}, k_{m_i})$ 的工作过程:首先计算 $I=((k_{m_i}+y)<<<k_{r_i})$,其次将 I 分成等长的四段,

即 $I=I_aI_bI_cI_d$，计算 $O=((S_1[I_a]\oplus S_2[I_b])-S_3[I_c]+S_4[I_d])$，即 O 为 f_1 的输出。

$f_2(y,k_{r_i},k_{m_i})$的工作过程：首先计算 $I=((k_{m_i}\oplus y)\lll k_{r_i})$，其次将 I 分成等长的四段，即 $I=I_aI_bI_cI_d$，计算 $O=((S_1[I_a]-S_2[I_b])+S_3[I_c]\oplus S_4[I_d])$，即 O 为 f_2 的输出。

$f_3(y,k_{r_i},k_{m_i})$的工作过程：首先计算 $I=((k_{m_i}-y)\lll k_{r_i})$，其次将 I 分成等长的四段，即 $I=I_aI_bI_cI_d$，计算 $O=((S_1[I_a]+S_2[I_b])\oplus S_3[I_c]-S_4[I_d])$，即 O 为 f_3 的输出。

这里 y 的长度为 32 比特，k_{r_i}的长度为 5 比特，称之为循环密钥(rotation key)，k_{m_i}的长度为 32 比特，称之为掩盖密钥(masking key)。$S_i(i=1,2,3,4)$是 4 个固定的 S-盒，限于篇幅这里没有列出。

解密过程：解密过程和加密过程完全一样，只是以逆序的方式使用密钥。用加密密钥导出解密密钥的过程如下：

$$\text{for } (i=0;i<12;i++)$$
$$k_{r_{new}}^{i}=k_r^{(11-i)}$$
$$k_{m_{new}}^{i}=k_m^{(11-i)}$$

其中 $k_r^{(i)}=\{k_{r_0}^{(i)},k_{r_1}^{(i)},k_{r_2}^{(i)},k_{r_3}^{(i)}\}$表示第 i 个四-轮的循环密钥之集，$k_m^{(i)}=\{k_{m_0}^{(i)},k_{m_1}^{(i)},k_{m_2}^{(i)},k_{m_3}^{(i)}\}$表示第 i 个四-轮的掩盖密钥之集。

密钥方案：*CAST*-256 算法的密钥方案由 256 比特的种子密钥 K 导出。整个密钥方案分两步来生成。

第一步：初始化。置 $c_m=2^{30}\sqrt{2}=5A827999$(十六进制)，$m_m=2^{30}\sqrt{3}=6ED9EBA1$(十六进制)，$c_r=19$，$m_r=17$。

$$\text{for } (i=0;i<24;i++)$$
$$\text{for } (j=0;j<8;j++)$$
$$t_{m_j}^{(i)}=c_m$$
$$c_m=(c_m+m_m)\bmod 2^{32}$$
$$t_{r_j}^{(i)}=c_r$$
$$c_r=(c_r+m_r)\bmod 32$$

第二步：生成密钥方案。置 $k=ABCDEFGH=$“256 比特的种子密钥 K”。

$$\text{for } (i=0;i<12;i++)$$
$$k\leftarrow\overline{\omega}_{2i}(k)$$
$$k\leftarrow\overline{\omega}_{2i-1}(k)$$
$$k_r^{(i)}\leftarrow k$$
$$k_m^{(i)}\leftarrow k$$

其中“$k\leftarrow\overline{\omega}_i(k)$”是下列赋值过程的简写(设 $k=ABCDEFGH$ 是一个 256 比特的组，A，B，...，H 的长度均为 32 比特)：

$$G=G\oplus f_1(H,t_{r_0}^{(i)},t_{m_0}^{(i)}),\ F=F\oplus f_2(G,t_{r_1}^{(i)},t_{m_1}^{(i)})$$
$$E=E\oplus f_3(F,t_{r_2}^{(i)},t_{m_2}^{(i)}),\ D=D\oplus f_1(E,t_{r_3}^{(i)},t_{m_3}^{(i)})$$
$$C=C\oplus f_2(D,t_{r_4}^{(i)},t_{m_4}^{(i)}),\ B=B\oplus f_3(C,t_{r_5}^{(i)},t_{m_5}^{(i)})$$
$$A=A\oplus f_1(B,t_{r_6}^{(i)},t_{m_6}^{(i)}),\ H=H\oplus f_2(A,t_{r_7}^{(i)},t_{m_7}^{(i)})$$

"$k_r^{(i)} \leftarrow k$"是下列赋值过程的简写：

$$k_{r_0}^{(i)} = 5\mathrm{LSB}(A) \qquad k_{r_1}^{(i)} = 5\mathrm{LSB}(C) \qquad k_{r_2}^{(i)} = 5\mathrm{LSB}(E) \qquad k_{r_3}^{(i)} = 5\mathrm{LSB}(G)$$

这里 5LSB(x)表示 x 的 5 比特低位。

"$k_m^{(i)} \leftarrow k$"是下列赋值过程的简写：

$$k_{m_0}^{(i)} = H \qquad k_{m_1}^{(i)} = F \qquad k_{m_2}^{(i)} = D \qquad k_{m_3}^{(i)} = B$$

2.2 CRYPTON

CRYPTON 算法是一个分组长度为 128 比特，密钥长度为 $64+32k(0 \leqslant k \leqslant 6)$比特，轮数为 r(建议取 $r=12$)的分组密码算法。该算法中主要使用了以下几种变换：

(1)非线性替换 γ。

为了实现非线性变换，CRYPTON 使用了两个 S-盒即 S_0 和 S_1。这两个 S-盒是从三个 F_2^4 上的置换使用一个 3-轮 Feistel 结构构作出来的，并满足：对任何 8 比特长的数 x，有 $S_0(S_1(x))=S_1(S_0(x))=x$。这三个置换分别为

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P_0	15	9	6	8	9	9	4	12	6	2	6	10	1	3	5	15
P_1	10	15	4	7	5	2	14	6	9	3	12	8	13	1	11	0
P_2	0	4	8	4	2	15	8	13	1	1	15	7	2	11	14	15

S_0 和 S_1 的构作过程如下：设 $x=x_1x_r$，x_1 和 x_r 均为 4 比特长，置 $y_r=x_r \oplus P_1(x_1 \oplus P_0(x_r))$，$y_1=x_1 \oplus P_0(x_r) \oplus P_2(y_r)$，$y=y_1y_r$。定义 $S_0(x)=y$，$S_1(x)$定义为 $S_0(x)^{-1}$。

在 CRYPTON 算法中交替使用了两个不同的变换 γ，在奇数轮中使用 γ_o，在偶数轮中使用 γ_e。变换 γ 是 4×4 字节矩阵上的一个逐字节替换。变换 γ_o 和 γ_e 满足条件：对任何 4×4 字节矩阵 A，有 $\gamma_o(\gamma_e(A))=\gamma_e(\gamma_o(A))=A$ (γ_o 和 γ_e 互逆)。这样做的目的是为了使加、解密过程相同。γ_o 和 γ_e 的变换过程如下：

$$\begin{array}{c|cccc} A[0] & a_{03} & a_{02} & a_{01} & a_{00} \\ A[1] & a_{13} & a_{12} & a_{11} & a_{10} \\ A[2] & a_{23} & a_{22} & a_{21} & a_{20} \\ A[3] & a_{33} & a_{32} & a_{31} & a_{30} \end{array} \xrightarrow{\gamma_o} \begin{array}{c|cccc} B[0] & S_1(a_{03}) & S_0(a_{02}) & S_1(a_{01}) & S_0(a_{00}) \\ B[1] & S_0(a_{13}) & S_1(a_{12}) & S_0(a_{11}) & S_1(a_{10}) \\ B[2] & S_1(a_{23}) & S_0(a_{22}) & S_1(a_{21}) & S_0(a_{20}) \\ B[3] & S_0(a_{33}) & S_1(a_{32}) & S_0(a_{31}) & S_1(a_{30}) \end{array}$$

$$\begin{array}{c|cccc} A[0] & a_{03} & a_{02} & a_{01} & a_{00} \\ A[1] & a_{13} & a_{12} & a_{11} & a_{10} \\ A[2] & a_{23} & a_{22} & a_{21} & a_{20} \\ A[3] & a_{33} & a_{32} & a_{31} & a_{30} \end{array} \xrightarrow{\gamma_e} \begin{array}{c|cccc} B[0] & S_0(a_{03}) & S_1(a_{02}) & S_0(a_{01}) & S_1(a_{00}) \\ B[1] & S_1(a_{13}) & S_0(a_{12}) & S_1(a_{11}) & S_0(a_{10}) \\ B[2] & S_0(a_{23}) & S_1(a_{22}) & S_0(a_{21}) & S_1(a_{20}) \\ B[3] & S_1(a_{33}) & S_0(a_{32}) & S_1(a_{31}) & S_0(a_{30}) \end{array}$$

(2)线性变换 π 和 τ。

π 是一个逐比特置换，在奇数轮中使用 π_o，在偶数轮中使用 π_e。二者之间满足关系：$\pi_e((A[3],A[2],A[1],A[0])')=\pi_o((A[2],A[1],A[0],A[3])')$，其中$(A[3],A[2],A[1],A[0])'$ 表示$(A[3],A[2],A[1],A[0])$的转置。$B=\pi_o(A)$和 $B=\pi_e(A)$分别定义为：

$$B[0] \leftarrow (A[0] \wedge M_0) \oplus (A[1] \wedge M_1) \oplus (A[2] \wedge M_2) \oplus (A[3] \wedge M_3)$$

$$B[1] \leftarrow (A[0] \wedge M_1) \oplus (A[1] \wedge M_2) \oplus (A[2] \wedge M_3) \oplus (A[3] \wedge M_0)$$
$$B[2] \leftarrow (A[0] \wedge M_2) \oplus (A[1] \wedge M_3) \oplus (A[2] \wedge M_0) \oplus (A[3] \wedge M_1)$$
$$B[3] \leftarrow (A[0] \wedge M_3) \oplus (A[1] \wedge M_0) \oplus (A[2] \wedge M_1) \oplus (A[3] \wedge M_2)$$

$$B[0] \leftarrow (A[0] \wedge M_1) \oplus (A[1] \wedge M_2) \oplus (A[2] \wedge M_3) \oplus (A[3] \wedge M_0)$$
$$B[1] \leftarrow (A[0] \wedge M_2) \oplus (A[1] \wedge M_3) \oplus (A[2] \wedge M_0) \oplus (A[3] \wedge M_1)$$
$$B[2] \leftarrow (A[0] \wedge M_3) \oplus (A[1] \wedge M_0) \oplus (A[2] \wedge M_1) \oplus (A[3] \wedge M_2)$$
$$B[3] \leftarrow (A[0] \wedge M_0) \oplus (A[1] \wedge M_1) \oplus (A[2] \wedge M_2) \oplus (A[3] \wedge M_3)$$

其中“$\wedge$”表示逐比特逻辑与，M_3，M_2，M_1 和 M_0 是四个掩盖字(每个字为 32 比特长)，M_3 = cff3fc3f，M_2 = f3fc3fcf，M_1 = fc3fcff3，M_0 = 3fcff3fc，这些数均为十六进制数。

π_o 的逆变换 $\pi_o^{-1}=\pi_o$，π_e 的逆变换 $\pi_e^{-1}=\pi_e$。

τ：$B=\tau(A)$是一个字节转置，这里 $b_{ij}=a_{ji}$。即

$$\begin{array}{l|llll} A[0] & a_{03} & a_{02} & a_{01} & a_{00} \\ A[1] & a_{13} & a_{12} & a_{11} & a_{10} \\ A[2] & a_{23} & a_{22} & a_{21} & a_{20} \\ A[3] & a_{33} & a_{32} & a_{31} & a_{30} \end{array} \xrightarrow{\ \tau\ } \begin{array}{l|llll} B[0] & a_{30} & a_{20} & a_{10} & a_{00} \\ B[1] & a_{31} & a_{21} & a_{11} & a_{01} \\ B[2] & a_{32} & a_{22} & a_{12} & a_{02} \\ B[3] & a_{33} & a_{23} & a_{13} & a_{03} \end{array}$$

τ 的逆变换 $\tau^{-1}=\tau$。

(3)密钥加变换 σ。

密钥加变换 σ_K：$B=\sigma_K(A)$通过轮密钥和数据字的异或运算来完成密钥的混合，其定义为

$$B[0] \leftarrow A[0] \oplus K[0]$$
$$B[1] \leftarrow A[1] \oplus K[1]$$
$$B[2] \leftarrow A[2] \oplus K[2]$$
$$B[3] \leftarrow A[3] \oplus K[3]$$

密钥加变换 σ_K 的逆变换 $\sigma_K^{-1}=\sigma_K$。

(4)轮变换 ρ。

CRYPTON 算法的轮变换由

下的加密变换为

$$E_K = \phi_e \circ \rho_{eK_e^r} \circ \rho_{oK_e^{r-1}} \circ \cdots \circ \rho_{eK_e^2} \circ \rho_{oK_e^1} \circ \sigma_{K_e^0}$$

其中 $\phi_e = \tau \circ \pi_e \circ \tau$。

解密过程：解密变换 D_K 为

$$D_K = \phi_e \circ \rho_{eK_d^r} \circ \rho_{oK_d^{r-1}} \circ \cdots \circ \rho_{eK_d^2} \circ \rho_{oK_d^1} \circ \sigma_{K_d^0}$$

其中解密轮密钥定义为

$$K_d^{r-i} = \begin{cases} \phi_e(K_e^i) & i = 0,2,4,\cdots \\ \phi_o(K_e^i) & i = 1,3,5,\cdots \end{cases} \qquad \phi_o = \tau \circ \pi_o \circ \tau$$

密钥方案：r-轮 CRYPTON 算法总共需要 $4(r+1)$个 32 比特的轮密钥，这些密钥从一个长度为 $64+32k(0 \leqslant k \leqslant 6)$比特的用户密钥生成。生成过程分两步来完成，第一步生成扩展密钥，第二步生成加、解密密钥。

扩展密钥的生成：设 $K = k_{u-1}k_{u-2}\cdots k_1k_0$ 是 $u(u=8+4i, i=0,1,\cdots,6)$字节长的用户密钥。首先在 K 的左边添加若干个零使得其长度为 256 比特。其次将其分成 8 个 32 比特字 $U[i](0 \leqslant i \leqslant 7)$：$U[i] = k_{4i+3}k_{4i+2}k_{4i+1}k_{4i}$。最后按下列办法计算 8 个扩展密钥$E_e[i]$：

$$(V_e[3],V_e[2],V_e[1],V_e[0])^t = (\tau \circ \gamma_o \circ \sigma_P \circ \pi_o)\ ((U[6],U[4],U[2],U[0])^t)$$

$$(V_e[7],V_e[6],V_e[5],V_e[4])^t = (\tau \circ \gamma_e \circ \sigma_Q \circ \pi_e)\ ((U[7],U[5],U[3],U[1])^t)$$

$$T_0 = V_e[3] \oplus V_e[2] \oplus V_e[1] \oplus V_e[0]$$

$$T_1 = V_e[7] \oplus V_e[6] \oplus V_e[5] \oplus V_e[4]$$

$$E_e[i] = V_e[i] \oplus T_1 \qquad i = 0,1,2,3$$

$$E_e[i] = V_e[i] \oplus T_0 \qquad i = 4,5,6,7$$

其中 $P = (P_3,P_2,P_1,P_0)^t$，$Q = (Q_3,Q_2,Q_1,Q_0)^t$，$P_3 = 510e527f$，$P_2 = a54ff53a$，$P_1 = 3c6ef372$，$P_0 = bb67ae85$，$Q_3 = cbbb9d5d$，$Q_2 = 5be0cd19$，$Q_1 = 1f83d9ab$，$Q_0 = 9b05688c$，这些数均为十六进制数。

加密轮密钥的生成：设 $K_e^i = (K_e[4i+3],K_e[4i+2],K_e[4i+1],K_e[4i])^t$ 是第 i 个轮密钥。将前 4 个扩展密钥赋给 K_e^0，将后 4 个扩展密钥赋给 K_e^1。轮密钥 K_e^{2i+2} 和 $K_e^{2i+3}(i \geqslant 0)$分别由 K_e^{2i} 和 K_e^{2i+1} 通过常数加和循环获得。更精确地说，在 K_e^{2i} 中的 4 个轮密钥中的两个通过左循环(循环次数是 8 的倍数)获得 K_e^{2i+2} 中的两个密钥，K_e^{2i} 中的剩余的两个密钥通过与常数相加获得 K_e^{2i+2} 中的另两个密钥。附表 2.2.1 给出了循环量和相加的常数。其中 $RC_0 = 01010101$，$RC_1 = 02020202$，$RC_2 = 04040404$，$RC_3 = 08080808$，$RC_4 = 10101010$，$RC_5 = 20202020$，这些数都是十六进制数。

例如，由 K_e^0 获得 K_e^2 的过程如下：

$$K_e[8] = K_e[0] \lll 8$$

$$K_e[9] = K_e[1] \oplus RC_0$$

$$K_e[10] = K_e[2] \lll 16$$

$$K_e[11] = K_e[3] \oplus RC_0$$

附表 2.2.1

	$K_r^{2i}\to K_r^{2i+2}$				$K_r^{2i+1}\to K_r^{2i+3}$			
i	3	2	1	0	3	2	1	0
0	RC_0	16	RC_0	8	24	RC_0	16	RC_0
1	8	RC_1	24	RC_1	RC_1	16	RC_1	8
2	RC_2	24	RC_2	16	8	RC_2	24	RC_2
3	16	RC_3	8	RC_3	RC_3	24	RC_3	16
4	RC_4	8	RC_4	24	16	RC_4	8	RC_4
5	24	RC_5	16	RC_5				

2.3 E2

E2 是一个分组长度为 128 比特，密钥长度为 128/192/256 比特的分组密码算法，该算法的主体是一个 12 轮的 Feistel 结构，在加密的开始和最后各用了一个变换，防止分析中的剥皮。在此算法中规定

$$B=F_2^8 \qquad W=B^4 \qquad H=B^8$$

加密过程：加密过程由初始变换 IT、12 轮的 Feistel 结构和末尾变换 FT 组成。$k_1,k_2,\ldots,k_{16}$是由种子密钥 K 生成的子密钥。令 M 是 128 比特的明文，首先计算 $M'=IT(M,k_{12},k_{14})$，然后把 M' 分成 L_0 和 R_0 两个长度为 64 的比特串。对 $r(1\leq r\leq 12)$，做以下操作：

$$R_r=L_{r-1}\oplus F(R_{r-1},k_r)$$
$$L_r=R_{r-1}$$

令 $C'=(R_{12},L_{12})$，最后，计算密文 $C=FT(C',k_{16},k_{15})$。

其中 F 是轮函数。

$IT(M,A,B)=BP((X\oplus A)\otimes B)$ $\qquad$ $FT(X,A,B)=(BP^{-1}(X)OB)\oplus A$

运算⊗定义如下：

$$X=Y\otimes B\ (X,Y,B\in F_2^{128})$$

其中$(y_1,y_2,y_3,y_4)=Y$ $(y_i\in F_2^{32},1\leqslant i\leqslant 4)$，$(b_1,b_2,b_3,b_4)=B$ $(b_i\in F_2^{32},1\leqslant i\leqslant 4)$，$x_i=y_i(b_i\vee 1)\bmod 2^{32}(1\leqslant i\leqslant 4)$，$X=(x_1,x_2,x_3,x_4)$，$\vee 1$ 表示用 $1\in Z_{2^{32}}$作比特逻辑或运算。

运算 O 定义如下：

$$X=YOB\ (X,Y,B\in F_2^{128})$$

其中$(y_1,y_2,y_3,y_4)=Y$ $(y_i\in F_2^{32},1\leqslant i\leqslant 4)$，$(b_1,b_2,b_3,b_4)=B$ $(b_i\in F_2^{32},1\leqslant i\leqslant 4)$，$x_i=y_i(b_i\vee 1)^{-1}\bmod 2^{32}(1\leqslant i\leqslant 4)$，$X=(x_1,x_2,x_3,x_4)$。

$$BP:W^4\to W^4,(X_1,X_2,X_3,X_4)\to(Y_1,Y_2,Y_3,Y_4)$$

其中，$X_i=(X_{i,1},X_{i,2},X_{i,3},X_{i,4})$ $(X_{ij}\in B,i=1,2,3,4,j=1,2,3,4)$，$Y_i=(X_{i,1},X_{i+1,2},X_{i+2,3},X_{i+3,4})(i=1,2,3,4)$，$(X_{i+4,j}=X_{i,j},i=1,2,3,4,j=1,2,3,4)$。

$$F:H\times H^2\to H \qquad (X,(K_1,K_2))\to Y=BRL(S(P(S(X\oplus K_1))\oplus K_2))$$
$$BRL:H\to H \qquad (b_1,b_2,b_3,\ldots,b_8)\to(b_2,b_3,\ldots,b_8,b_1)$$
$$S:H\to H \qquad (x_1,x_2,\ldots,x_8)\to(s(x_1),s(x_2),\ldots,s(x_8))$$

$s:B\to B$ $\qquad$ $x\to s(x)=97x^{127}+225$，先在 F_{2^8}上运算 x^{127}，将其结果记为 t，然后在 Z 上运算 $97\times t+225$ 即为 $s(x)$。

$$P:H\to H,(x_1,x_2,\ldots,x_8)\to(x_1,x_2,\ldots,x_8)P$$

其中

$$P=\begin{bmatrix}0&1&1&1&1&1&1&0\\1&0&1&1&0&1&1&1\\1&1&0&1&1&0&1&1\\1&1&1&0&1&1&0&1\\1&1&0&1&1&1&0&0\\1&1&1&0&0&1&1&0\\0&1&1&1&0&0&1&1\\1&0&1&1&1&0&0&1\end{bmatrix}$$

解密过程：$E2$ 的解密过程和加密过程类似，仅仅是子密钥顺序不同。加密过程的子密钥顺序为$(k_{13},k_{14},k_1,k_2,k_3,k_4,k_5,k_6,k_7,k_8,k_9,k_{10},k_{11},k_{12},k_{15},k_{16})$，解密过程的子密钥顺序为$(k_{16},k_{15},k_{12},k_{11},k_{10},k_9,k_8,k_7,k_6,k_5,k_4,k_3,k_2,k_1,k_{14},k_{13})$。

密钥方案：首先定义函数 G 如下：

$$G:H^4\times H\to H^4\times(H^4\times H)$$

$$((X_1,X_2,X_3,X_4),U_0)\to((U_1,U_2,U_3,U_4),((Y_1,Y_2,Y_3,Y_4),V))$$

其中 $Y_i=f(X_i)\ (i=1,2,3,4)$，$U_i=f(U_{i-1})\oplus Y_i(i=1,2,3,4)$，$V=U_4$，$f:H\to H$，$X\to P(S(X))$。

设 $K=(K_1,K_2,K_3,K_4)\in H^4$ 为种子密钥(当种子密钥为 128 比特时，取 $K_3=S(S(S(v_1)))$，$K_4=S(S(S(S(v_1))))$。当种子密钥为 192 比特时，取 $K_4=S(S(S(S(v_1))))$。我们用下面的方法构作 16 个子密钥：

$$v_1=0123456789abcdef(\text{十六进制})$$

$$(L_0,(Y_0,v_0))=G(K,v_1)$$

$$(L_{i+1},(Y_{i+1},v_{i+1}))=G(Y_i,v_i)\quad(i=0,1,2,\ldots,7)$$

$$(l_{4i},l_{4i+1},l_{4i+2},l_{4i+3})=L_{i+1}\quad(i=0,1,\ldots,7)$$

$$(t_i^{(0)},t_i^{(1)},\ldots,t_i^{(7)})=l_i\quad(i=0,1,\ldots,31)$$

$$k=(t_{0+(i\bmod 2)}^{[i/2]},t_{2+(i\bmod 2)}^{[i/2]},\ldots,t_{30+(i\bmod 2)}^{[i/2]})\qquad(i=0,1,\ldots,15)$$

其中，$L_i,Y_i\in H^4$，$l_i,v_i\in H$，$t_i^{(j)}\in B$。

2.4 DEAL

DEAL 是基于 DES 的一个分组密码算法，DEAL 的分组长度为 128 比特，密钥长度为 128/192/256 比特。DEAL 是一个 $r(r=6,8)$轮的 Feistel 结构。

加密过程：令 $P=(X_0^L,X_0^R)$为 128 比特的明文，对 $j=1,2,\ldots,r$，作下列的计算：

$$X_j^L=X_{j-1}^R\oplus DES_{RK_j}(X_{j-1}^L)$$

$$X_j^R=X_{j-1}^L$$

$$C=(X_r^L,X_r^R)\text{ 为密文}$$

其中 $DES_K(X)$表示在密钥 K 控制下 DES 对 X 的加密值。

解密过程：解密过程和加密过程类似，仅仅是子密钥顺序的不同。

密钥方案：令 $K=0123456789abcdef$(十六进制)，其中(i)是一个从 0 记起的 64 比特

序数串，该串中只设置第 i-1 个位置其余位置全设置为 0，例如“8000000000000000”(十六进制)表示〈1〉。

当种子密钥(K_1,K_2)为 128 比特时，其中 K_i 为 64 比特，子密钥用下述方法构作：

$$RK_1=DES_K(K_1),RK_2=DES_K(K_2\oplus RK_1),RK_3=DES_K(K_1\oplus\langle 1\rangle\oplus RK_2)$$

$$RK_4=DES_K(K_2\oplus\langle 2\rangle\oplus RK_3),RK_5=DES_K(K_1\oplus\langle 4\rangle\oplus RK_4),$$

$$K_6=DES_K(K_2\oplus\langle 8\rangle\oplus RK_5)$$

当种子密钥(K_1,K_2,K_3)为 192 比特时，其中 K_i 为 64 比特，子密钥用下述方法构作：

$$RK_1=DES_K(K_1),RK_2=DES_K(K_2\oplus RK_1)$$

$$RK_3=DES_K(K_3\oplus RK_2),RK_4=DES_K(K_1\oplus\langle 1\rangle\oplus RK_3)$$

$$RK_5=DES_K(K_2\oplus\langle 2\rangle\oplus RK_4),RK_6=DES_K(K_3\oplus\langle 4\rangle\oplus RK_5)$$

当种子密钥(K_1,K_2,K_3,K_4)为 256 比特时，其中 K_i 为 64 比特，子密钥用下述方法构作：

$$RK_1=DES_K(K_1),RK_2=DES_K(K_2\oplus RK_1),RK_3$$
$$=DES_K(K_3\oplus RK_2),RK_4=DES_K(K_4\oplus RK_3)$$

$$RK_5=DES_K(K_1\oplus\langle 1\rangle\oplus RK_4),RK_6=DES_K(K_2\oplus\langle 2\rangle\oplus RK_5)$$

$$RK_7=DES_K(K_3\oplus\langle 4\rangle\oplus RK_6),RK_8=DES_K(K_4\oplus\langle 8\rangle\oplus RK_7)$$

2.5 FROG

FROG 是一种非正规结构的密码，其基本设计思想是通过内部密钥隐藏大多数计算过程，也就是尽可能不给攻击者对实际执行过程有更多的了解，从而挫败任何攻击。FROG 的分组长度为 128 比特，密钥长度是可变的，从 5 个字节到 125 个字节。

加密过程：FROG 采用 8 轮迭代，每轮需要一组长度为 288 字节的内部密钥，表示为 XorBuf $=(A_0,A_1,\ldots,A_{15})$(16 字节)，Substpermu(256 字节)和 bombpermu $=(B_0,B_1,\ldots,B_{15})$(16 字节)。设 $X=(X_0,\ldots,X_{15})$是 16 字节输入，每次迭代依次移动 16 字节(从 0 到 15)，并在每个字节上执行下列 4 步运算。令 $Y=(Y_0,\ldots,Y_{15})$是对第 i 个字节操作后所得的输出，则对第 $i+1$ 个字节的操作如下：

(1)用 Y 的第 $i+1$ 个字节与 XorBuf 的第 $i+1$ 个字节进行异或运算。$Y_0,\ldots,Y_{i-1},Y_i\oplus A_i,\ldots,Y_{15}$。

(2)把第一步的运算结果 $Y_i\oplus A_i$，作 Substpermu 变换，即

$$Y_0,\ldots,Y_{i-1},\text{Substpermu}(Y_i\oplus A_i),\ldots,Y_{15}$$

(3)给第 $i+2$ 个字节异或加上第 $i+1$ 个字节$((i+1)\bmod 16)$，即

$$Y_0,\ldots,Y_{i-1},\text{Substpermu}(Y_i\oplus A_i),Y_{i+1}\oplus\text{Substpermu}(Y_i\oplus A_i),\ldots,Y_{15}$$

(4)给第 b_i 个字节异或加上 Substpermu$(Y_i\oplus A_i)$，b_i 表示 bombpermu 的第 $i+1$ 个字节对应的整数，即

$$Y_0,\ldots,Y_{i-1},\text{Substpermu}(Y_i\oplus A_i),Y_{i+1}\oplus$$
$$\text{Substpermu}(Y_i\oplus A_i),\ldots,Y_{bi}\oplus\text{Substpermu}(Y_i\oplus A_i),\ldots,Y_{15}$$

解密过程：解密过程是加密过程的逆。

密钥方案:FROG 的加密过程需要 2304 字节内部子密钥,种子密钥是可变长的,范围是从 5 字节到 125 字节。FROG 的密钥设置过程是一种递归过程,首先 FROG 建立一个简单的内部密钥,然后,该简单的内部密钥用 FROG 将内部密钥译成 CBC(密文块链接)模式,以产生最后的正式内部密钥。

2.6 SAFER+

SAFER+是基于 SAFER 系列算法提出的,因此,它的安全性可以说是经过了时间的考验。另外,SAFER+算法中仅使用了字节运算,并且所需的内存小,因此在 Smart 卡等方面的应用是很有优势的。SAFER+算法是一个代换/线性交换密码,它的加密/解密过程不相似。

加密过程:16 字节的明文经过 $r(8,12,16)$轮加密,每一轮均使用两个 16 字节的子密钥,子密钥$(K_1,K_2,\ldots,K_{2r})$由种子密钥 K 经过密钥扩展算法来产生。最后,将子密钥 K_{2r+1}与第 r 轮加密后所产生的块相"加",即得 16 字节的密文。这里的"加"是指第 1,4,5,8,9,12,13 和 16 字节逐位模加,第 2,3,6,7,10,11,14 和 15 字节模 256 加。

第 i 轮加密:将子密钥 K_{2i-1}按如下方式"加"到 16 字节的输入上去:第 1,4,5,8,9,12,13 和 16 字节逐比特模 2 加,第 2,3,6,7,10,11,14 和 15 字节模 256 加。所得 16 字节经过一个非线性层:第 j 字节 x 变换为 $45^x \bmod 257$,$j=1,4,5,8,9,12,13,16$;第 j 字节 x 变换为 $\log_{45}(x)$,$j=2,3,6,7,10,11,14,15$。然后将子密钥 K_{2i}"加"到非线性层的输出上去;第 2,3,6,7,10,11,14,和 15 逐比特模 2 加,第 1,4,5,8,9,12,13 和 16 字节模 256 加。所得 16 字节的结果为

$$x=[x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,x_9,x_{10},x_{11},x_{12},x_{13},x_{14},x_{15},x_{16}]$$

模 256 右乘矩阵 M 产生 16 字节的轮输出

$$y=[y_1,y_2,y_3,y_4,y_5,y_6,y_7,y_8,y_9,y_{10},y_{11},y_{12},y_{13},y_{14},y_{15},y_{16}]$$

即 $y=xM$,其中 M 是 16×16 的矩阵,如下定义:

$$\begin{bmatrix}
2 & 2 & 1 & 1 & 16 & 8 & 2 & 1 & 4 & 2 & 4 & 2 & 1 & 1 & 4 & 4\\
1 & 1 & 1 & 1 & 8 & 4 & 2 & 1 & 2 & 1 & 4 & 2 & 1 & 1 & 2 & 2\\
1 & 1 & 4 & 4 & 2 & 1 & 4 & 2 & 4 & 2 & 16 & 8 & 2 & 2 & 1 & 1\\
1 & 1 & 2 & 2 & 2 & 1 & 2 & 1 & 4 & 2 & 8 & 4 & 1 & 1 & 1 & 1\\
4 & 4 & 2 & 1 & 4 & 2 & 4 & 2 & 16 & 8 & 1 & 1 & 1 & 1 & 2 & 2\\
2 & 2 & 2 & 1 & 2 & 1 & 4 & 2 & 8 & 4 & 1 & 1 & 1 & 1 & 1 & 1\\
1 & 1 & 4 & 2 & 4 & 2 & 16 & 8 & 2 & 1 & 2 & 2 & 4 & 4 & 1 & 1\\
1 & 1 & 2 & 1 & 4 & 2 & 8 & 4 & 2 & 1 & 1 & 1 & 2 & 2 & 1 & 1\\
2 & 1 & 16 & 8 & 1 & 1 & 2 & 2 & 1 & 1 & 4 & 4 & 4 & 2 & 4 & 2\\
2 & 1 & 8 & 4 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 4 & 2 & 2 & 1\\
4 & 2 & 4 & 2 & 4 & 4 & 1 & 1 & 2 & 2 & 1 & 1 & 16 & 8 & 2 & 1\\
1 & 1 & 4 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 8 & 4 & 2 & 1\\
4 & 2 & 2 & 2 & 1 & 1 & 4 & 4 & 1 & 1 & 4 & 2 & 2 & 1 & 16 & 8\\
4 & 2 & 1 & 1 & 1 & 1 & 2 & 2 & 1 & 1 & 2 & 1 & 2 & 1 & 8 & 4\\
16 & 8 & 1 & 1 & 2 & 2 & 1 & 1 & 4 & 4 & 2 & 1 & 4 & 2 & 4 & 2\\
8 & 4 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 1 & 2 & 1 & 4 & 2
\end{bmatrix}$$

解密过程：解密过程只是加密过程的逆。

密钥方案：在密钥扩展算法中需要 32 个 128 比特的常量 $B_2, B_3, \ldots, B_{33}, B_i = (B_{i,1}, \ldots, B_{i,j}, \ldots, B_{i,16})$，字节 $B_{i,j}$ 如下计算：

$$B_{i,j} = 45^{(45^{17i+j} \bmod 257)} \bmod 257 \quad (i = 2,3,\ldots,17, j = 1,2,\ldots,16)$$

$$B_{i,j} = 45^{17i+j} \bmod 257 \quad (i = 18,19,\ldots,33, j = 1,2,\ldots,16)$$

K 为 128 比特的密钥扩展方案：K 作为第一个子密钥 K_1 并放入 17 个字节的寄存器的前 16 个字节位置，寄存器的最后一个位置放入 K 的 16 字节的模 2 加。然后寄存器的每个字节循环左移 3 比特，将 16 字节的常量 B_2 分别与寄存器的第 2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17 字节逐模 256 加即得子密钥 K_2。寄存器的每个字节再循环左移 3 比特，将 16 字节的常量 B_3 分别与寄存器的第 3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,1 字节逐字节模 256 加即得子密钥 K_3。如此下去，直到将 16 字节的常量 B_{17} 分别与寄存器的第 17,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 字节逐字节模 256 加，即得子密钥 K_{17}。

K 为 192 比特的密钥扩展方案：K 的前 16 字节作为第一个子密钥 K_1 并将 K 放入 25 个字节的寄存器的前 24 个字节位置，寄存器的最后一个位置放入 K 的 24 个字节的模 2 加。然后寄存器的每个字节循环左移 3 比特，将 16 字节的常量 B_2 分别与寄存器的第 2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17 字节逐字节模 256 加即得子密钥 K_2。寄存器的每个字节再循环左移 3 比特，将 16 字节的常量 B_3 分别与寄存器的第 3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18 字节逐字节模 256 加即得子密钥 K_3。如此下去，直到将 16 字节的常量 B_{25} 分别与寄存器的第 25,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 字节逐字节模 256 加，即得子密钥 K_{25}。

K 为 256 比特的密钥扩展方案：K 的前 16 字节作为第一个子密钥 K_1 并将 K 放入 33 个字节的寄存器的前 32 个字节位置，寄存器的最后一个位置放入 K 的 32 个字节的模 2 加。然后寄存器的每个字节循环左移 3 比特，将 16 字节的常量 B_2 分别与寄存器的第 2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17 字节逐字节模 256 加即得子密钥 K_2。寄存器的每个字节再循环左移 3 比特，将 16 字节的常量 B_3 分别与寄存器的第 3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18 字节逐字节模 256 加即得子密钥 K_3。如此下去，直到将 16 字节的常量 B_{33} 分别与寄存器的第 33,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 字节逐字节模 256 加，即得子密钥 K_{33}。

2.7 RC6

RC6 是在 RC5 的基础上设计的，众所周知，RC5 是一个非常简洁的算法，它的特点是大量使用数据依赖循环。RC6 继承了这些优点，为了满足 NIST 的要求，即分组长度为 128 比特，RC6 使用了 4 个寄存器，并加进 32 比特的整数乘法，用于加强扩散特性。RC6 更精确的表示是 RC6-$w/r/b$，其中字长为 w 比特，r 为加密轮数，b 为加密密钥用字节表示的长度。这里规定 $w=32, r=20, b=16(24,32)$。RC6 用了下面几种基本运算：(1)整数模 2^w 加和减，分别表示为“+”和“−”。(2)比特字的逐位模 2 加，表示为“⊕”。(3)整数模 2^w 乘，表示为“×”。(4)左右循环，用 $a \lll b$ 表示将 w 比特的字 a 向左循环 b 位，用 $a \ggg b$

表示将 w 比特的字 a 向左循环 b 位。

加密过程：把 128 比特明文放入 4 个 32 比特的寄存器 A,B,C,D。

$$
\begin{aligned}
&B = B + S[0]\\
&D = D + S[1]\\
&\text{For } i = 1 \text{ to } r \text{ do}\\
&\quad t = (B \times (2B+1)) \lll \lg w\\
&\quad u = (D \times (2D+1)) \lll \lg w\\
&\quad A = ((A \oplus t) \lll u) + S[2i]\\
&\quad C = ((C \oplus u) \lll t) + S[2i+1]\\
&\quad (A,B,C,D) = (B,C,D,A)\\
&A = A + S[2r+2]\\
&C = C + S[2r+3]
\end{aligned}
$$

(A,B,C,D)即为密文。

解密过程：把 128 比特密文放入 4 个 32 比特的寄存器 A,B,C,D。

$$
\begin{aligned}
&C = C - S[2r+3]\\
&A = A - S[2r+2]\\
&\text{For } i = r \text{ down to } 1 \text{ do}\\
&\quad (A,B,C,D) = (D,A,B,C)\\
&\quad u = (D \times (2D+1)) \lll \lg w\\
&\quad t = (B \times (2B+1)) \lll \lg w\\
&\quad t = (B \times (2B+1)) \lll \lg w\\
&\quad C = ((C - S[2i+1]) \ggg t) \oplus u\\
&\quad A = ((A - S[2i]) \ggg u) \oplus t\\
&\quad D = D - S[1]\\
&\quad B = B - S[0]
\end{aligned}
$$

(A,B,C,D)即为明文。

密钥方案：密钥扩展方案类似于 RC5 的密钥扩展方案。在密钥扩展中用到了两个常数 P_{32}和 Q_{32}，$P_{32}=B7E15163$(十六进制)，$Q_{32}=9E3779B9$(十六进制)。首先，将种子密钥 K 输入 c 个 w 比特字的 $L[0],\ldots,L[c-1]$阵列，若不够，用 0 字节填充。

$$
\begin{aligned}
&S[0] = P_w\\
&\text{For } i = 1 \text{ to } 2r+3 \text{ do}\\
&\quad S[i] = S[i-1] + Q_w\\
&\quad A = B = i = j = 0\\
&\quad v = 3 \times \max\{c, 2r+4\}\\
&\text{For } s = 1 \text{ to } v \text{ do}\\
&\quad A = S[i] = (S[i] + A + B) \lll 3\\
&\quad B = L[j] = (L[j] + A + B) \lll (A+B)\\
&\quad i = (i+1) \bmod (2r+4)\\
&\quad j = (j+1) \bmod c
\end{aligned}
$$

输出 $S[0],S[1],\ldots,S[2r+3]$即为子密钥。

2.8 MAGENTA

MAGENTA 算法的核心部分是以快速哈达马变换为基础，算法的设计思想是应用一些可在软件和硬件中有效实现的简单且透明的技术。

令 α 为域 F_{2^8}的本原元，其生成多项式为 $p(x)=x^8+x^6+x^5+x^2+1$，且 $p(\alpha)=0$。令 B 表示 8 比特 0、1 序列的集合。我们通过$(x_7,x_6,\ldots,x_0)\to 2^7x_7+2^6x_6+\ldots+x_0$，将 B 中的元素和$\{0,1,\ldots,255\}$中的整数联系起来，则对 $x\in B$，定义

$$f(x)=\begin{cases}0 & x=255\\ \alpha^x & x\neq 255\end{cases}$$

对$(x,y)\in B^2$，定义

$$A(x,y)=f(x\oplus f(y))$$

$$PE(x,y)=(A(x,y),A(y,x))=(f(x\oplus f(y)),f(y\oplus f(x)))$$

对$(x_0,x_1,\ldots,x_{15})\in B^{16}$，定义

$$T(x_0,x_1,\ldots,x_{15})=\prod(\prod(\prod(\prod(x_0,x_1,\ldots,x_{15}))))$$

其中

$$\prod(x_0,x_1,\ldots,x_{15})=(PE(x_0,x_8),PE(x_1,x_9),\ldots,PE(x_7,x_{15}))$$

对 $X=(x_0,x_1,\ldots,x_{15})\in B^{16}$，定义

$$X_e=(x_0,x_2,\ldots,x_{14})$$

$$X_0=(x_1,x_3,\ldots,x_{15})$$

对 $j\geq 1$，定义

$$C^{(j+1)}(x_0,\ldots,x_{15})=T((x_0,\ldots,x_7)\oplus C_e^{(j)},(x_8,\ldots,x_{15})\oplus C_o^{(j)})$$

其中 $C^{(1)}=T(x_0,\ldots,x_{15})$。

令 $E^{(r)}(x_0,\ldots,x_{15})=C_e^{(r)}$。

加密过程：MAGENTA 采用的是 6 或 8 轮的 Feistel 结构，它的每一轮变换如下定义：

$$F_k(x_0,\ldots,x_{15})=((x_8,\ldots,x_{15}),(x_0,\ldots,x_7)\oplus E^{(3)}(x_8,\ldots,x_{15},y_0,\ldots,y_7))$$

其中 $X=(x_0,x_1,\ldots,x_{15})$是 128 比特的输入，$K=(k_0,\ldots,k_7)$是 64 比特的密钥。

MAGENTA 算法支持使用三种密钥规模，即

128 比特：$K=(K_1,K_2)$

192 比特：$K=(K_1,K_2,K_3)$

256 比特：$K=(K_1,K_2,K_3,K_4)$

MAGENTA 的加密算法可以用下面的函数表示：

$$\mathrm{Enc}_K(M)=\begin{cases}F_{K1}(F_{K1}(F_{K2}(F_{K2}(F_{K1}(F_{K1}(M))))))) & K=(K_1,K_2)\\ F_{K1}(F_{K2}(F_{K3}(F_{K3}(F_{K2}(F_{K1}(M.)))))) & K=(K_1,K_2,K_3)\\ F_{K1}(F_{K2}(F_{K3}(F_{K4}(F_{K4}(F_{K3}(F_{K2}(F_{K1}(M)))))))) & K=(K_1,K_2,K_3,K_4)\end{cases}$$

解密过程：解密和加密类似，可以用下面的函数表示：

$$\mathrm{Dec}_K(M) = V(\mathrm{Enc}_K(V(M)))$$

其中 $V((x_0,\ldots,x_{15})=(x_8,x_9,\ldots,x_{15},x_0,\ldots,x_7)$。

2.9 LOKI97

LOKI97 是 LOKI89 和 LOKI91 的进一步改进。它的分组长度为 128 比特，密钥长度为 128/192/256 比特。它采用的是 Feistel 结构。

加密过程：设 $P=L_0|R_0$ 为 128 比特的明文输入，用下列方式计算密文，对 $j=1,\ldots,16$

$$R_i = L_{i-1} \oplus f(R_{i-1} + K_{3i-2}, K_{3i-1})$$
$$L_i = R_{i-1} + K_{3i-2} + K_{3i}$$

$C=R_{16}|L_{16}$为密文。

轮函数 f：

$$F_2^{64} \times F_2^{64} \to F_2^{64}$$
$$\mathrm{f}(A,B) = Sb(P(Sa(E(KP(A,B)))),B)$$

KP(A,B)是一个简单的密钥控制置换。它将 64 比特输入 A 分成两个 32 比特字，用输入 B 的较低(最右边)32 比特确定是交换这两 个字中对应位置的比特(如果密钥比特是 1)，还是不交换(如果密钥比特为 0)。

E 是一个扩展函数，它从一个 64 比特输入产生一个 96 比特输出。

[4 − 0,63 − 56|58 − 48|52 − 40|42 − 32|34 − 24|28 − 16|18 − 8|12 − 0]

S_a 由盒 S_1 和盒 S_2 并置构成，$S_a=[S_1,S_2,S_1,S_2,S_2,S_1,S_2,S_1]$，$S_a$ 的输入是的 E 输出。

P 把输入比特[63−0]映射到输出比特：

[56,48,40,32,24,16,08,00,57,49,41,33,25,17,09,01
58,50,42,34,26,18,10,02,59,51,43,35,27,19,11,03
60,52,44,36,28,20,12,04,61,53,45,37,29,21,13,05
62,54,46,38,30,22,14,06,63,55,47,39,31,23,15,07]

即输入比特 63 转入输出比特 56，输入比特 62 转入输出比特 48 等。

S_b 由盒 S_1 和盒 S_2 并置构成，$S_b=[S_2,S_2,S_1,S_1,S_2,S_2,S_1,S_1]$，$S_b$ 的输入是 B[63−61]|P[63−56]，B[60−58]|P[55−48]，B[57−53]|P[47−40]，B[52−48]|P[39−32]，B[47−45]|P[31−24]，B[44−42]|P[23−16]，B[41−37]|P[15−8]，B[36−32]|P[7−0]。其中 B[63−61]表示 B 的第 63 到 61 比特组成的比特串。Sb 的第一个 S 盒 S_2 的输入为 B[63−61]|P[63−56]。

解密过程：输入密文 $C=R_{16}|L_{16}$，然后反向对轮进行操作，即对 $i=16,\ldots,1$

$$L_{i-1} = R_i \oplus f(L_i - K_{3i}, K_{3i-1})$$
$$R_{i-1} = L_i - K_{3i} - K_{3i-2}$$

$P=L_0|R_0$ 即为明文。

密钥方案：16 轮 LOKI97 需要 48 个 64 比特的密钥。我们用下述方法把种子密钥 K 扩展为子密钥。首先，依据种子密钥的长度，预置 4 个 64 比特字[$K4_0|K3_0|K2_0|K1_0$]。

$K=[Ka|Kb|Kc|Kd]$ 为 256 比特，令 $[K4_0|K3_0|K2_0|K1_0]=[Ka|Kb|Kc|Kd]$

$K=[Ka|Kb|Kc]$ 为 192 比特，令 $[K4_0|K3_0|K2_0|K1_0]=[Ka|Kb|Kc|f(Ka,Kb)]$

$K=[Ka|Kb]$ 为 128 比特，令 $[K4_0|K3_0|K2_0|K1_0]=[Ka|Kb|f(Kb,Ka)|f(Ka,Kb)]$

然后对 $i=1,\ldots,48$，做如下计算：

$K_i=K1_i=K4_{i-1}\oplus g_i(K1_{i-1},K3_{i-1},K2_{i-1}),K4_i=K3_{i-1},K3_i=K2_{i-1},K2_i=K1_{i-1}$

其中

$$g_i(K1,K3,K2)=f(K1+K3+(\text{Delta}*i),K2)$$

$$\text{Delta}=[(\text{sqrt}(5)-1)*2^{63}]=9E3779B97F4A7C15\ (\text{十六进制})$$

2.10 SERPENT

SERPENT 采用了和 DES 类似的 S-盒，不过它用了一种新的结构，此结构保证 SERPENT的雪崩效应及快速实现，设计者已证明 SERPENT 能抵抗已知的所有攻击，并且声称 SERPENT 比三重 DES 更安全。SERPENT 的分组长度为 128 比特，种子密钥为 128/192/256 比特。

加密过程：SERPENT 的加密过程由三部分组成：第一部分是初始置换；第二部分是 32 轮的加密操作，每一轮包含密钥混合运算、S-盒及线性变换；第三部分是末尾置换。

一个 128 比特的明文 P，首先经过初始置换，然后在 33 个子密钥 $(K_0,\ldots,K_{32})$ 控制下经过 33 轮加密，最后经过末尾置换，即得密文 C。

(1)初始置换 IP。

0	32	64	96	1	33	65	97	2	34	66	98	3	35	67	99
4	36	68	100	5	37	69	101	6	38	70	102	7	39	71	103
8	40	72	104	9	41	73	105	10	42	74	106	11	43	75	107
12	44	76	108	13	45	77	109	14	46	78	110	15	47	79	111
16	48	80	112	17	49	81	113	18	50	82	114	19	51	83	115
20	52	84	116	21	53	85	117	22	54	86	118	23	55	87	119
24	56	88	120	25	57	89	121	26	58	90	122	27	59	91	123
28	60	92	124	29	61	93	125	30	62	94	126	31	63	95	127

(2)32 轮的加密操作。

令 $B_0=IP(P)$ 为第 1 轮的输入，F_i 表示第 $i+1$ 轮的加密函数，B_i 为第 i 轮的输出。32 轮的加密操作可用下列式子表示：

$$B_0=IP(P)$$

$$B_{i+1}=F_i(B_i)$$

$$C=FP(B_{32})$$

其中 $F_i(X)=L(\overline{S}_i(X\oplus K_i)),i=0,1,\ldots,30$。$F_i(X)=\overline{S}_i(X\oplus K_i)\oplus K_{32}\quad i=31$。$\overline{S}_i$ 是 32 个 S_j 盒的并置，即 $\overline{S}_i=(S_jS_j..S_j)$。其中 $j=i\bmod 8$。

Serpent 使用了 8 个 F_2^4 上的 S-盒，它们分别是：

S_0:	3	8	15	1	10	6	5	11	14	13	4	2	7	0	9	12
S_1:	15	12	2	7	9	0	5	10	1	11	14	8	6	13	3	4
S_2:	8	6	7	9	3	12	10	15	13	1	14	4	0	11	5	2
S_3:	0	15	11	8	12	9	6	3	13	1	2	4	10	7	5	14
S_4:	1	15	8	3	12	0	11	6	2	5	4	10	9	14	7	13
S_5:	15	5	2	11	4	10	9	12	0	3	14	8	13	6	7	1
S_6:	7	2	12	5	8	4	6	11	14	9	1	15	13	3	10	0
S_7:	1	13	15	0	14	8	2	11	7	4	12	10	9	3	5	6

线性变换 L(这里的线性性是指对每个字而言即对每个字作线性变换)如下定义:设$(X_0, X_1, X_2, X_3)=S_i(B_i \oplus K_i)$,置

$$
\begin{aligned}
X_0 &= X_0 \lll 13 \\
X_2 &= X_2 \lll 3 \\
X_1 &= X_1 \oplus X_0 \oplus X_2 \\
X_3 &= X_3 \oplus X_2 \oplus (X_0 \ll 3) \\
X_1 &= X_1 \lll 1 \\
X_3 &= X_3 \lll 7 \\
X_0 &= X_0 \oplus X_1 \oplus X_3 \\
X_2 &= X_2 \oplus X_3 \oplus (X_1 \ll 7) \\
X_0 &= X_0 \lll 5 \\
X_2 &= X_2 \lll 22 \\
B_{i+1} &= (X_0, X_1, X_2, X_3)
\end{aligned}
$$

其中$\lll$表示左循环,$\ll$表示移位。

(3)末尾置换 $FP=IP^{-1}$。

解密过程:解密过程是加密过程的逆。

密钥方案:首先,把种子密钥 K 填充为 256 比特,并表示为 8 个 32 比特的字 $w_{-8},\ldots,w_{-1}$,然后利用 下列的递归式:

$$w_i = (w_{i-8} \oplus w_{i-5} \oplus w_{i-3} \oplus w_{i-1} \oplus \Phi \oplus i) \lll 11$$

可得 132 个 32 比特的字 $w_0,\ldots,w_{131}$。最后利用 S 盒获得子密钥

$K_0 = S_3(w_0, w_1, w_2, w_3)$　　$K_1 = S_2(w_4, w_5, w_6, w_7)$　　$K_2 = S_1(w_8, w_9, w_{10}, w_{11})$

$K_3 = S_0(w_{12}, w_{13}, w_{14}, w_{15})$　　$K_4 = S_7(w_{16}, w_{17}, w_{18}, w_{19})$　……

$K_i = S_j(w_{4i}, w_{4i+1}, w_{4i+2}, w_{4i+3})$　　$(i+j) \bmod 8 = 3$　……

$K_{33} = S_2(w_{128}, w_{129}, w_{130}, w_{131})$

其中 Φ=9e3779b9(十六进制)。

2.11 MARS

Mars 是 IBM 公司提供的一个候选算法,它的特点是充分使用非平衡的 Feistel 网络。为了保证加密和解密的强度相当,Mars 用结构类似的两部分组成。该算法是面向字运算

的，所有内部操作均以 32 比特字为单位。

加密过程：加密过程由六部分组成。第一部分是密钥加(按 32 比特字加)，第二部分是不受密钥控制的 8 轮前期混合运算，第三部分是密钥控制下的 8 轮前期加密变换，第四部分是密钥控制下的 8 轮后期加密变换，第五部分是不受密钥控制的 8 轮后期混合运算，第六部分是密钥减(按 32 比特字减)。

(1)前期混合运算。

对 128 比特输入 $X=(X_3,X_2,X_1,X_0)$进行下面的运算：

设 $X_0=(X_{0,3},X_{0,2},X_{0,1},X_{0,0})$

置 $X_1=X_1\oplus S_0[X_{0,0}]+S_1[X_{0,1}]$　$X_2=X_2+S_0[X_{0,2}]$　$X_3=X_3\oplus S_1[X_{0,3}]$

$X_0=X_3+ROR(X_0,24)$

设 $X_1=(X_{1,3},X_{1,2},X_{1,1},X_{1,0})$

置 $X_2=X_2\oplus S_0[X_{1,0}]+S_1[X_{1,1}]$　$X_3=X_3+S_0[X_{1,2}]$　$X_0=X_0\oplus S_1[X_{1,3}]$

$X_1=X_2+ROR(X_1,24)$

设 $X_2=(X_{2,3},X_{2,2},X_{2,1},X_{2,0})$

置 $X_3=X_3\oplus S_0[X_{2,0}]+S_1[X_{2,1}]$　$X_1=X_1+S_1[X_{2,3}]$　$X_0=X_0\oplus S_0[X_{2,2}]$

$X_2=X_2+ROR(X_2,24)$

设 $X_3=(X_{3,3},X_{3,2},X_{3,1},X_{3,0})$

置 $X_0=X_0\oplus S_0[X_{3,0}]+S_1[X_{3,1}]$　$X_1=X_1+S_0[X_{3,2}]$　$X_2=X_2\oplus S_1[X_{3,3}]$

$X_3=X_3+ROR(X_3,24)$

重复上述过程一次。其中 $ROR(X,n)$表示 X 右循环 n 比特。

(2)前期加密变换。

首先，我们给出 E 函数的定义，E 函数是 Mars 的核心，它的输入是 32 比特的字 X，输出是 3 个 32 比特的字(Y_2,Y_1,Y_0)，即

$$Y_2=ROL(ROL(X,13)\times k',10)$$

$$Y_1=ROL[(X+k),ROR(Y_2,5)]$$

$$Y_0=ROL(S[(X+k)\text{ 的低 9 比特}]\oplus ROR(Y_2,5)\oplus Y_2,Y_2)$$

其中 S 是从 F_2^9 到 F_2^{32} 的盒子，k 和 k'(要求 k'是奇数)是子密钥，$ROL(X,n)$表示 X 左循环 n 比特。

前期加密变换由 8 轮 Feistel 类型 3 网络组成，第 i 轮的输入为 4 个 32 比特字 $X=(X_3,X_2,X_1,X_0)$，输出 $Y=(Y_3,Y_2,Y_1,Y_0)$由下列式子计算：

$$Y_3=ROL(X_0,13)$$

$$Y_2=X_3\oplus E(X_0)_2$$

$$Y_1=X_2+E(X_0)_1$$

$$Y_0=X_1+E(X_0)_0$$

其中 $E(X)_i$ 表示 $E(X)$的第 i 个 32 比特字，$S[X]$表示用 X 作指针代替 S-盒中的第 X 个元素。

(3)后期加密变换。

后期加密变换也由 8 轮 Feistel 类型 3 网络组成，第 i 轮的输入为 4 个 32 比特字 $X=(X_3,X_2,X_1,X_0)$，输出 $Y=(Y_3,Y_2,Y_1,Y_0)$由下列式子计算：

$$Y_3 = ROL(X_0,13)$$
$$Y_2 = X_3 + E(X_0)_0$$
$$Y_1 = X_2 + E(X_0)_1$$
$$Y_0 = X_1 \oplus E(X_0)_2$$

(4)后期混合运算。

对 128 比特输入 $X=(X_3,X_2,X_1,X_0)$进行下面的运算：

设 $X_0=(X_{0,3},X_{0,2},X_{0,1},X_{0,0})$

置 $X_1=X_1\oplus S_1[X_{0,0}]$　$X_2=X_2-S_0[X_{0,1}]$　$X_3=X_3-S_1[X_{0,2}]\oplus S_0[X_{0,3}]$

$X_0=ROL(X_0,24)$

设 $X_1=(X_{1,3},X_{1,2},X_{1,1},X_{1,0})$

置 $X_2=X_2\oplus S_1[X_{1,0}]$　$X_3=X_3-S_0[X_{1,1}]$　$X_0=X_0-S_1[X_{1,2}]\oplus S_1[X_{1,3}]$

$X_1=ROL(X_1,24)$　$X_2=X_2-X_1$

设 $X_2=(X_{2,3},X_{2,2},X_{2,1},X_{2,0})$

置 $X_3=X_3\oplus S_1[X_{2,0}]$　$X_0=X_0-S_0[X_{2,1}]$　$X_1=X_1-S_1[X_{2,2}]\oplus S_0[X_{2,3}]$

$X_2=ROL(X_2,24)$　$X_3=X_3-X_0$

设 $X_3=(X_{3,3},X_{3,2},X_{3,1},X_{3,0})$

置 $X_0=X_0\oplus S_1[X_{3,0}]$　$X_1=X_1-S_0[X_{3,1}]$　$X_2=X_2-S_1[X_{3,2}]\oplus S_0[X_{3,3}]$

$X_3=ROL(X_3,24)$

重复上述过程一次。

解密方程：解密过程是加密过程的逆。

密钥方案：密钥扩展算法由下列 4 步构成：

(1)把种子密钥 K 分成 n 个 32 比特字 $K[0],K[1],\ldots,K[n-1]$，

对 $i=-7,\ldots,-1$　$T[i]=S[i+7]$

对 $i=0,\ldots,38$　$T[i]=((T[i-7]\oplus T[i-2])\lll 3)\oplus K[i \bmod n]\oplus i$

最后令 $T[39]=n$。

(2)重复 7 次下面的过程：

$T[i]=(T[i]+S[T[i-1]$ 的低 9 比特$])\lll 9$　$i=1,2,\ldots,39$

$T[0]=(T[0]+S[T[39]$ 的低 9 比特$])\lll 9$

(3)对 $T[.]$的顺序重组

$$K[7i \bmod 40]=T[i], i=0,1,\ldots,39$$

(4)对用于乘法的 16 个字进行检测，保证它们都不“弱”。详细的检测方法不再列出。

其中≪表示移位。

关于 S-盒：Mars 中的 S-盒采用伪随机方式生成，但需对其差分和线性特性进行检测，其生成过程如下：

$$S[5i+j]=\text{SHA-1}(i,c_1,c_2,c_3)_j \quad i=0,1,\ldots,102 \quad j=0,1,\ldots,4$$

其中 SHA-1(.)$_j$ 表示 SHA-1(.)输出的第 j 个字，$c_1=b7e15162$，$c_2=243f6a88$，c_3 变动，直到找到好的 S-盒为止。

2.12 Rijndael

Rijndael 算法的原形是 Square 算法，它的设计策略是宽轨迹策略(Wide Trail Strategy)。这种策略是针对差分分析和线性分析提出的。Rijndael 是一个迭代分组密码，其分组长度和密钥长度都是可变的，但是为了满足 AES 的要求，分组长度为 128 比特，密钥长度为 128/192/256 比特，相应的轮数 r 为 10/12/14。

加密过程：X 是 Rijndael 密码的 128 比特输入，Y 是 128 比特的输出，则 Rijndael 密码可用下式表示：

$$Y = O_{K_{r+1}} \circ T \circ \Gamma \circ O_{K_r} \circ \prod \circ T \circ \Gamma \circ O_{K_{r-1}} \circ \cdots\cdots \circ \prod \circ T \circ \Gamma \circ O_{K_1}(X)$$

其中“$\circ$”表示置换的复合，$K_1, K_2 .. K_{r+1}$是 $r+1$ 个子密钥。

$O_{K_i}: F_2^{128} \to F_2^{128}$ 是一个置换，对 $X \in F_2^{128}$

$$O_{K_i}(X) = X \oplus K_i$$

$T: F_2^{128} \to F_2^{128}$ 是一个置换，X 是 T 的输入，首先，把 X 分成 16 个字节，即

$$X = (X_{00}, X_{01}, X_{02}, X_{03}, X_{10}, X_{11}, X_{12}, X_{13}, X_{20}, X_{21}, X_{22}, X_{23}, X_{30}, X_{31}, X_{32}, X_{33})$$

输出

$$Y = T(X) = (X_{00}, X_{01}, X_{02}, X_{03}, X_{13}, X_{10}, X_{11}, X_{12}, X_{22}, X_{23}, X_{20}, X_{21}, X_{31}, X_{32}, X_{33}, X_{30})$$

$\prod: F_2^{128} \to F_2^{128}$ 是一个置换，X 是 T 的输入，首先，把 X 分成 16 个字节，即

$$X = (X_{00}, X_{01}, X_{02}, X_{03}, X_{10}, X_{11}, X_{12}, X_{13}, X_{20}, X_{21}, X_{22}, X_{23}, X_{30}, X_{31}, X_{32}, X_{33})$$

输出

$$Y = \prod(X) = (Y_{00}, Y_{01}, Y_{02}, Y_{03}, Y_{10}, Y_{11}, Y_{12}, Y_{13}, Y_{20}, Y_{21}, Y_{22}, Y_{23}, Y_{30}, Y_{31}, Y_{32}, Y_{33})$$

其中

$$\begin{bmatrix} Y_{0i} \\ Y_{1i} \\ Y_{2i} \\ Y_{3i} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} X_{0i} \\ X_{1i} \\ X_{2i} \\ X_{3i} \end{bmatrix}$$

$\Gamma: F_2^{128} \to F_2^{128}$ 是一个置换，它由 16 个 F_2^8 上的 S-盒并置构成，$S = L \circ F$。F 是有限域 F_{2^8} 上的乘法逆，即 $F(X) = X^{-1}$(约定 $F(0) = 0$)，$L(X) = AX + b$，

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, b = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

解密过程：解密过程是加密过程的逆，这里不再详述。

密钥方案：在加密过程中，需要 $r+1$ 个子密钥，需要构作 $4(r+1)$ 个 32 比特字。当种子密钥为 128 和 192 比特时，构作 $4(r+1)$ 个 32 比特字的程序是一样的，但当种子密钥为 256 比特时，用另一个不同的程序构作 $4(r+1)$ 个 32 比特字。限于篇幅，这里就不在详述。

2.13 DFC

DFC 是基于 Vaudenay 的抗相关攻击技术设计的，它的分组长度为 128 比特，密钥长度为 128/192/256 比特。

加密过程：DFC 是一个 8 轮的 Feistel 结构，设明文 $P=L_0R_0$，第 i 轮操作如下：

$$R_i = F_{K_i}(R_{i-1}) \oplus L_{i-1}$$

$$L_i = R_{i-1}$$

$C=(R_8,L_8)$是密文。

轮函数 F 如下定义：

$$F_{K_i}(X) = CP(((aX+b)\bmod(2^{64}+13))\bmod 2^{64})$$

其中 X 是 64 比特输入，$K_i=(a,b)$是子密钥。

CP 置换如下定义：

$$\mathrm{CP}(Y) = (((Y_r \oplus RT'(\mathrm{trunc}_6(Y_l))) \mid (Y_r \oplus KC) \mid KD \bmod 2^{64}) \bmod 2^{64}$$

其中 $Y=(Y_r,Y_l)$是 64 比特输入，trun $c_6(Y_l)$表示 Y_l 的前 6 比特组成的比特串，RT 是输入为 6 比特，输出为 32 比特的置换，它以表的形式给出，KC 是 32 比特的常数，KD 是 64 比特的常数。

解密过程：解密过程和加密过程类似。

密钥方案：首先，用常数 KS 将种子密钥 K 扩充到 256 比特，然后，将其分成 8 个 32 比特的字 $\mathrm{PK}_1,\ldots,\mathrm{PK}_8$。

令

$$\mathrm{OAP}_1 = \mathrm{PK}_1 \mid \mathrm{PK}_8 \qquad \mathrm{OBP}_1 = \mathrm{PK}_5 \mid \mathrm{PK}_4$$

$$\mathrm{EAP}_1 = \mathrm{PK}_2 \mid \mathrm{PK}_7 \qquad \mathrm{EBP}_1 = \mathrm{PK}_6 \mid \mathrm{PK}_3$$

对 $i=2,3,4$，定义：

$$\mathrm{OAP}_i = \mathrm{OAP}_1 \oplus \mathrm{KA}_i \qquad \mathrm{OBP}_i = \mathrm{OBP}_1 \oplus \mathrm{KB}_i$$

$$\mathrm{EAP}_i = \mathrm{EAP}_1 \oplus \mathrm{KA}_i \qquad \mathrm{EBP}_i = \mathrm{EBP}_1 \oplus \mathrm{KB}_i$$

其中 KA_i，KB_i 都是固定的常数。

令

$$\mathrm{RK}_0 = \mathrm{RV}_{1,0} \mid \mathrm{RV}_{1,1} = 0 \text{ (128 比特长)}$$

依下列式子计算 R_{1j}：

$$\mathrm{RV}_{i,j+1} = \begin{cases} F_{\mathrm{OAP}_j \mid \mathrm{OBP}_j}(\mathrm{RV}_{i,j}) \oplus \mathrm{RV}_{i,j-1} & i \text{ 为奇数} \\ F_{\mathrm{EAP}_j \mid \mathrm{EBP}_j}(\mathrm{RV}_{i,j}) \oplus \mathrm{RV}_{i,j-1} & i \text{ 为偶数} \end{cases} \tag{2.13.1}$$

令

$$\mathrm{RK}_1 = \mathrm{RV}_{1,5} \mid \mathrm{RV}_{1,4} = \mathrm{RV}_{2,0} \mid \mathrm{RV}_{2,1}$$

依式(2.13.1)，计算 $\mathrm{RV}_{2,j},\ldots,\mathrm{RV}_{8j}$的值，令 $\mathrm{RK}_i=\mathrm{RV}_{i,5} \mid \mathrm{RV}_{i,4}$为子密钥。

2.14 Twofish

Twofish 是一个分组长度为 128 比特，密钥长度为 128/192/256 比特的分组密码算法，它的总体结构是一个 16 轮的 Feistel 结构，主要特点是 S-盒由密钥控制。

加密过程：Twofish 的加密分三部分。第一部分是初始白化，即把 128 比特的明文分成 4 个 32 比特的字，然后从左到右，依次异或加 4 个 32 比特的子密钥 K_0, K_1, K_2, K_3。第二部分是 16 轮的加密，在第 i 轮中，令 $R_{i,0}, R_{i,1}, R_{i,2}, R_{i,3}$ 为 4 个 32 比特的输入，4 个 32 比特的输出 $R_{i+1,0}, R_{i+1,1}, R_{i+1,2}, R_{i+1,3}$ 用下列式子计算：

$$
\begin{aligned}
R_{i+1,0} &= [(g(R_{i,0}) + g(R_{i,1}\langle\langle\langle 8) + K_{2i+8}) \oplus R_{i,2}]\rangle\rangle\rangle 1\\
R_{i+1,1} &= (g(R_{i,0}) + 2g(R_{i,1}\langle\langle\langle 8) + K_{2i+9}) \oplus (R_{i,3}\langle\langle\langle 1)\\
R_{i+1,2} &= R_{i,0}\\
R_{i+1,3} &= R_{i,1}
\end{aligned}
$$

其中 $+$ 是模 2^{32} 加，$\langle\langle\langle 1$ 表示左循环 1 比特，$\rangle\rangle\rangle 1$ 表示右循环 1 比特。第三部分是末尾白化，即对第 16 轮的输出 $R_{17,0}, R_{17,1}, R_{17,2}, R_{17,3}$ 依次模 2 加子密钥 K_4, K_5, K_6, K_7，密文 $C = (R_{17,0} \oplus K_4, R_{17,1} \oplus K_5, R_{17,2} \oplus K_6, R_{17,3} \oplus K_7)$。

函数 g 是 Twofish 的核心，在 g 的构作中，用到两个固定的 F_2^8 上的置换 q_0 和 q_1，q_0 和 q_1 分别利用两组 4 个 F_2^4 上的置换 t_0, t_1, t_2, t_3 用下列方法构作：

$$
\begin{aligned}
a_0, b_0 &= [x/16], x \bmod 16\\
a_1 &= a_0 \oplus b_0\\
b_1 &= a_0 \oplus ROR(b_0, 1) \oplus 8a_0 \bmod 16\\
a_2, b_2 &= t_0[a_1], t_1[b_1]\\
a_3 &= a_2 \oplus b_2\\
b_3 &= a_2 \oplus ROR(b_2, 1) \oplus 8a_2 \bmod 16\\
a_4, b_4 &= t_2[a_3], t_3[b_3]\\
y &= 16b_4 + a_4
\end{aligned}
$$

其中 $ROR(b_0, 1)$ 表示 $b_0\rangle\rangle\rangle 1$。

q_0 所用的置换为

$$
\begin{aligned}
t_0 &= [8\ 1\ 7\ d\ 6\ f\ 3\ 2\ 0\ b\ 5\ 9\ e\ c\ a\ 4]\\
t_1 &= [e\ c\ b\ 8\ 1\ 2\ 3\ 5\ f\ 4\ a\ 6\ 7\ 0\ 9\ d]\\
t_2 &= [b\ a\ 5\ e\ 6\ d\ 9\ 0\ c\ 8\ f\ 3\ 2\ 4\ 7\ 1]\\
t_3 &= [d\ 7\ f\ 4\ 1\ 2\ 6\ e\ 9\ b\ 3\ 0\ 8\ 5\ c\ a]
\end{aligned}
$$

q_1 所用的置换为

$$
\begin{aligned}
t_0 &= [2\ 8\ b\ d\ f\ 7\ 6\ e\ 3\ 1\ 9\ 4\ 0\ a\ c\ 5]\\
t_1 &= [1\ e\ 2\ b\ 4\ c\ 3\ 7\ 6\ d\ a\ 5\ f\ 9\ 0\ 8]\\
t_2 &= [4\ c\ 7\ 5\ 1\ 6\ 9\ a\ 0\ e\ d\ 8\ 2\ b\ 3\ f]\\
t_3 &= [b\ 9\ 5\ 1\ c\ 3\ d\ e\ 6\ 4\ 7\ f\ 2\ 0\ 8\ a]
\end{aligned}
$$

令 $N = 128(192, 256)$ 是密钥长度，$k = N/64$，种子密钥由 $8k$ 个字节 $m_0, \ldots, m_{8k-1}$ 组成，令

$$M_i = (m_{4i}, m_{4i+1}, m_{4i+2}, m_{4i+3}) \quad (i = 0,1,\ldots,2k-1)$$
$$M_e = (M_0, M_2, \ldots, M_{2k-2})$$
$$M_o = (M_1, M_3, \ldots, M_{2k-1})$$

$$\begin{pmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \end{pmatrix} = \begin{pmatrix} 01 & a4 & 55 & 87 & 5a & 58 & db & 9e \\ a4 & 56 & 82 & f3 & 1e & c6 & 68 & e5 \\ 02 & a1 & fc & c1 & 47 & ae & 3d & 19 \\ a4 & 55 & 87 & 5a & 58 & db & 9e & 03 \end{pmatrix} \begin{pmatrix} m_{8i} \\ m_{8i+1} \\ m_{8i+2} \\ m_{8i+3} \\ m_{8i+4} \\ m_{8i+5} \\ m_{8i+6} \\ m_{8i+7} \end{pmatrix}$$

令

$$S_i = (s_{i,0}, s_{i,1}, s_{i,2}, s_{i,3})$$
$$S = (S_{k-1}, S_{k-2}, \ldots, S_0)$$

我们定义 $g(X)=h(X,S)$，其中函数 h 如下定义：

输入一个 32 比特字 X 和 k 个 32 比特字 $(L_0,\ldots,L_{k-1})$，输出为 32 比特字 Z。

首先，把输入的每个 32 比特字分成 4 个字节，即 $L_i=(l_{i,0},l_{i,1},l_{i,2},l_{i,3})$，$X=(x_0,x_1,x_2,x_3)$，令

$$y_{k,j} = x_j (j = 0,1,2,3)$$

当 $k=4$ 时

$$y_{3,0} = q_1[y_{4,0}] \oplus l_{3,0}$$
$$y_{3,1} = q_0[y_{4,1}] \oplus l_{3,1}$$
$$y_{3,2} = q_0[y_{4,2}] \oplus l_{3,2}$$
$$y_{3,3} = q_1[y_{4,3}] \oplus l_{3,3}$$

当 $k\geq 3$ 时

$$y_{2,0} = q_1[y_{3,0}] \oplus l_{2,0}$$
$$y_{2,1} = q_1[y_{2,1}] \oplus l_{2,1}$$
$$y_{2,2} = q_0[y_{2,2}] \oplus l_{2,2}$$
$$y_{2,3} = q_0[y_{2,3}] \oplus l_{2,3}$$

在任何情况下，

$$y_0 = q_1[q_0[q_0[y_{2,0}] \oplus l_{1,0}] \oplus l_{0,0}]$$
$$y_1 = q_0[q_0[q_1[y_{2,1}] \oplus l_{1,1}] \oplus l_{0,1}]$$
$$y_2 = q_1[q_1[q_0[y_{2,2}] \oplus l_{1,2}] \oplus l_{0,2}]$$
$$y_3 = q_0[q_1[q_1[y_{2,3}] \oplus l_{1,3}] \oplus l_{0,3}]$$

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 01 & ef & 5b & 5b \\ 5b & ef & ef & 01 \\ ef & 5b & 01 & ef \\ ef & 01 & ef & 5b \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

令 $h(X,S)=Z=(z_0,z_1,z_2,z_3)$。

解密过程：解密过程和加密过程类似。

密钥方案：

$$p=2^{24}+2^{16}+2^8+1$$
$$A_i=h(2ip,M_e)$$
$$B_i=ROL(h((2i+1)p,M_o),8)$$
$$K_{2i}=(A_i+B_i)\bmod 2^{32}$$
$$K_{2i+1}=ROL((A_i+2B_i)\bmod 2^{32},9)$$

其中 $ROL(X,n)$表示 $X\langle\langle\langle n$。

2.15 HPC

HPC 密码的分组长度和密钥长度都是任意的，0 比特或多个比特。HPC 密码有 512 比特可选的二级密钥即 SPICE，一个主密钥对每个二级密钥值给出一个不同的加密。HPC 密码由 5 个不同的子密码组成，究竟采用哪一个子密码是由需加密的消息的长度来决定（见表 2.15.1）。每个子密码使用自己的密钥扩展表，这些密钥扩展表通过密钥伪随机产生，每个表共有 256 个 64 比特字。所有的密钥扩展表都使用同一个算法，只有初始值不同。

表 2.15.1 5 个子密码

名称	加密长度(比特)
子密码 1 (HPC-Tiny)	0－35
子密码 2 (HPC-Short)	36－64
子密码 3 (HPC-Medium)	65－128
子密码 4 (HPC-Long)	129－512
子密码 5 (HPC-Extended)	513＋

HPC 密码的内部使用无符号的 64 比特字，任何可变的值比如密钥长度、明文长度或密文长度都以数个 64 比特字表示，可能后接一个适当调整的零碎字。SPICE 由 8 个字的数组指定。主要使用的操作有模加、模减、模乘、异或和移位。

从 AES 的征稿情况来看，目前设计分组密码仍然离不开旧的模式，无论在理论上还是在技术上都没有多大创新。如果读者想详细了解有关 AES 候选算法的最新情况，请去 NIST 主页观光，网址为：http://csrc.nist.gov/encryption/aes/aeshome.htm#candidates。

3 注记和文献

本附录的第一部分我们主要介绍了该书中所需要的一些必要的数学知识。限于篇幅，我们不可能作详细阐述，这里将介绍一些有关的参考文献以便读者在进一步阅读时参考。

有关概率方面的基础知识，可参阅文献[1,2,3]。有关数论方面的基础知识，可参阅文献[4,5]。有关数论算法，可参阅文献[6]。有关数论与密码的文献有文献[7]。有关抽象代数方面的基础知识可参阅文献[8,9,10]。有关有限域方面的知识，可参阅文献[11,12]。另外，有一些代数编码书中也对有限域作了很好的介绍，参阅文献[13,14,15,16]。

本附录的第二部分我们简要介绍了美国国家标准技术研究所(NIST)最近公布的十五个 AES 候选算法的基本思想。无论算法好坏,我们对此没有作任何评价。

参 考 文 献

[1] Feller, W., An Introduction to Probability Theory and Its Applications, John Wiley & Sons, New York, 3ed edition, 1968.

[2] 概率论,复旦大学编,人民教育出版社,北京,1979.

[3] 概率论与数理统计,华东师范大学数学系编,高等教育出版社,北京,1983.

[4] H. M. 维诺拉陀夫著,裘光明译,数论基础,高等教育出版社,北京,1956.

[5] 华罗庚,数论导引,科学出版社,北京,1975.

[6] *Bach*, *E.*, *Shallit*, J., Algorithmic Number Theory, The MIT Press, Cambridge, Massachusetts, London, England 1996.

[7] Loxtor, J. H., Number Theory and Cryptography, Cambridge University Press, World *Publishing Corp*, 1990.

[8] Jacobson, N., Basic Algebra, San Francisco: W. H. Freeman and Company, 1979.

[9] 冯克勤,李尚志,查建国,近世代数引论,中国科学技术大学出版社,合肥,1998.

[10] 张禾瑞,近世代数基础,高等教育出版社,北京,1978.

[11] Lidl, R., Niederreiter N., Finite Fields, Cambridge University Press, Cambridge, 1984.

[12] Mceliece, R. J., Finite Fields for Computer Scientists and Engineers, Kluwer Academic Publishers, Boston, 1987.

[13] Berlekamp, E. R., Algebraic Coding Theory, New York: McGraw-Hill, 1968.

[14] Macwilliams, F. J. Sloane, N. J. A., The Theory of *Error-correcting Codes*, North Holland Publishing Company 1977.

[15] 万哲先,代数和编码,科学出版社,北京,1985.

[16] 肖国镇,卿斯汉,编码理论,国防工业出版社,北京,1993.